The realisation of CAD/CAM/CNC interoperability in prismatic part manufacturing

Nassehi, Aydin

Award date:
2007

Awarding institution:
University of Bath

The Realisation of CAD/CAM/CNC Interoperability in Prismatic Part Manufacturing

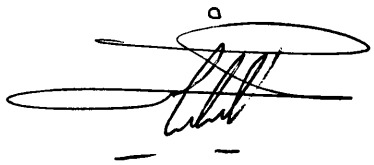
Aydin Nassehi

A thesis submitted for the degree of Doctor of Philosophy

University of Bath

Department of Mechanical Engineering

June 2007



ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

Dedicated to Maman and Pedar

Table of Contents

Acknowledgements v
Abstract vi
List of Abbreviations vii
Definitions ix
List of Figures xi
List of Tables xiv
1. Introduction 1
2. Research Aims, Scope and Context 4
2.1. Introduction 4
2.2. Research Aims 4
2.3. Research Methodology and Objectives 6
2.4. Research Context and Boundaries 7
2.4.1. Interoperability and Adaptability in Manufacturing Integration 7
2.4.2. Prismatic Parts and Milling Technology in CAx 8
2.4.3. STEP-NC (ISO14649) in Standards 8
2.4.4. Application of Information Technology 9
2.5. The Scope of the Research 9
2.5.1. Review of the State-of-the-Art in CAD/CAM/CNC Integration 9
2.5.2. Specification of a Novel Framework for Realisation of Interoperability 9
2.5.3. Defining Manufacturing Resource Abstraction 9
2.5.4. Investigation of Utilising a Comprehensive Manufacturing Lexicon 10
2.5.5. Exploration of Mobile Agents as Information Carriers 10
2.5.6. Realisation of a Prototype Interoperable CAx Framework Implementation 10
2.5.7. Evaluation of the Interoperable CAx Framework Prototype 10
3. State-of-the-Art in CAD/CAM/CNC Integration 11
3.1. Introduction 11
3.2. The CAx Chain: A Short History and Background 11
3.3. Integration Standards 14
3.3.1. IGES, VDA-FS, SET 14
3.3.2. The Standard for the Exchange of Product Data Model (STEP) 15
3.3.3. STEP-NC 21
3.3.4. Knowledge Interchange Format (KIF) 22
3.3.5. Extensible Mark-up Language (XML) 22
3.3.6. Common Object Request Broker Architecture (CORBA) 22
3.3.7. Web Services 23
3.3.8. Knowledge Query and Manipulation Language (KQML) 23
3.4. CAx Interoperability: The State-of-the-Art 23
3.4.1. STEP-AP238 and ISO14649 Based Solutions 23
3.4.2. Agent Based Manufacturing Interoperability 37
3.4.3. Computer Aided Process Planning in Relation to Interoperability 40
3.4.4. Manufacturing Interoperability Based on Enterprise Integration 41

3.4.5. Semantic Interoperability 44
3.4.6. Open and Intelligent CNC for Interoperability 47
3.4.7. Remote Web-Based Control and Monitoring for Integration of CNCs 47
3.4.8. Manufacturing Interoperability by Low-Level Data Transformation 48
3.5. Critique 49
3.5.1. The Requirement for Modification of Current Resources 49
3.5.2. Lack of Semantic Interoperability in CAx Manufacturing 52
3.5.3. Reactive Research as Opposed to Proactive / Revolutionary Research 52
3.5.4. STEP-NC and Object-Orientation 52
4. A Novel Object-Oriented Interoperable CAx Framework 54
4.1. Introduction 54
4.2. Requirements for an Interoperable CAx Framework 54
4.2.1. Standardisation 54
4.2.2. The Ability to Use Existing Resources without Modification 56
4.2.3. Object-Orientation 56
4.2.4. Semantic Interoperability 57
4.3. Framework Functionalities 57
4.4. Resource Abstraction 59
4.5. Information Models: Creating a Manufacturing Lexicon 59
4.6. Information Transfer Mechanisms 61
4.7. An Overall View of the Interoperable CAx Framework 62
5. CAx Resource Abstraction 65
5.1. Introduction 65
5.2. Information Flow in the Prismatic CAx Chain 65
5.2.1. Information Flow in the State-Of-The-Art CAx Chain 65
5.2.2. Information Flow in a STEP-NC Compliant CAx Chain 66
5.3. CAx Resource Abstraction 67
5.4. CAX Resource Abstractor Object 71
5.5. External Definition of CAx Resources 74
5.6. Summary 77
6. A Computational Platform for a Comprehensive CAx Manufacturing Lexicon 78
6.1. Introduction 78
6.2. Layers of Manufacturing Information 78
6.3. STEP-NC, the Interlingua for Interoperability 80
6.4. The Integrated Platform for Process Planning and Control (IP^3AC) 81
6.4.1. Entities 84
6.4.2. Types 86
6.4.3. Constants 88
6.4.4. Functions 88
6.4.5. Where, Derive, Inverse 88
6.4.6. Optional 89
6.5. EXPRESS Translator 89
6.6. Advantages and Disadvantages of IP^3AC 90
6.6.1. Simple Manipulation of Manufacturing Information in the Java Environment 91
6.6.2. The Ability to Define Additional Methods during Automatic Code Generation 91
6.6.3. The Ability to Utilise Existing Libraries to Manipulate Manufacturing Data 92
6.6.4. Extensibility to Provide a Foundation for Manufacturing Software 93
6.7. Summary 94

7. Information Exchange Using Mobile Agents in the CAx Chain 95
7.1. Introduction 95
7.2. Agents as Carriers of Information 95
7.3. Information Storage in Agents 97
7.4. Agent Mobility, Persistence and Security 98
7.5. A Mobile Agent Based Information Transfer Framework 101
7.6. Advantages of Using Agents for Information Transfer 106
7.6.1. Monitoring 107
7.6.2. Data Collection from a Variety of Data Sources 108
7.6.3. Contingency In Case Of Failed Transfers and Semantic Integrity 108
7.6.4. Decrease in Bandwidth Requirements 108
7.7. Summary 109
8. Realisation of Interoperable CAx 110
8.1. Introduction 110
8.2. Development of the Data Structures Using IP^{3}AC 111
8.3. Development of the CAx Resource Abstractors 113
8.3.1. The Interoperable CAD/CAM System 113
8.3.2. Resource Abstraction for the Siemens 840D CNC 116
8.3.3. Resource Abstraction for the Heidenhain iTNC530 CNC 118
8.4. Development of Information Transfer Procedure 120
8.5. Prototype Integration 122
8.6. Summary 122
9. Evaluation of the Interoperable CAx Framework Prototype 123
9.1. Introduction 123
9.2. Evaluation Methodology 123
9.3. Test Component Design 124
9.4. Test Component Programming On CNC 1 with Siemens 840D Controller 130
9.5. Generation of the Standardised Manufacturing Information 132
9.6. Generation of Proprietary Code for Heidenhain iTNC530 133
9.7. Results of the Evaluation 135
10. Discussion 136
10.1. Introduction 136
10.2. State-Of-The-Art in CAD/CAM/CNC Integration 136
10.3. A Novel Framework for Realisation of Interoperability 137
10.3.1. Manufacturing Resource Abstraction 138
10.3.2. Encoding Manufacturing Information 138
10.3.3. Utilisation of Mobile Agents to Provide Communications 139
10.3.4. Prototype Implementation 139
10.4. Evaluation of the Prototype Using a Test Component 140
10.5. Advantages of Implementation of the Interoperable CAx Framework 141
10.6. Limitations of the Interoperable CAx Framework 143
11. Conclusions and Future Work 144
11.1. Introduction 144
11.2. Conclusions 144
11.3. Contributions to Knowledge 146
11.4. Future Work 147
11.4.1. Integration of the STEP-NC Research 147

11.4.2. The Role of the Research in Realising the Manufuture Vision 147
11.4.3. The Application of Service Oriented Architecture 148
11.4.4. Utilisation of the Semantic Web 149
11.4.5. Comprehensive Manufacturing Resource Models 149
11.4.6. Towards a Universal Manufacturing Platform 149
References 151
Appendix A. Publications Based on the Research 163
A.1. Journal Articles 163
A.2. Conference Papers 166
Appendix B. Program Listings for the Test Component 168
B.1. ShopMill MPF Listing 168
B.2. STEP-NC listing 171
B.3. Heidenhain Listing 186

Acknowledgements

The work documented in this thesis would not have been possible without the valuable support of many people.

First and foremost I would like to thank my supervisor, **Professor Stephen Newman** for giving me the opportunity to explore the most interesting areas of the research while making sure that I completed doing everything that was necessary. I thank you sir, not only for the invaluable support and advice that you have provided me, both in work and life, but also for consistently and constantly setting standards that have made me strive to do better.

I would also like to express my gratitude to **Dr. Richard Allen** whose advice and guidance was pivotal in helping me find my research direction when I felt lost.

I should also thank **Professor Safaieh** and **Dr. Ali Reza Haji** who made it possible for me to continue my studies in the UK as well as **Mammad Agha Esmailee** and **Leyla Khanoom** for putting up with me during the many years of my education.

My friends **Mehrad, Reza, Souroush, Marzi, Kiarash, Neda, Marta, Teresa, Bryan** and **Bernardo** helped me transform my potentially isolated life as a PhD student into a rich and fulfilling experience and for this I can never thank them enough.

Glavije, you gave me the inspiration and the drive that kept me going through the difficult bits of this journey. I thank you for providing me with the dream; and also for your support with your unrivalled drawing and measurement skills.

And last but not least I want to thank my brother. **Ramin** you have been a pillar of support, a wise friend, a man of interesting words and most importantly a mischievous accomplice for doing extraordinary things and living to laugh about them.

Abstract

The manufacturing industry today, is the culmination of paradigms and trends that have roots in the industrial revolution. With the ongoing information revolution, these paradigms and trends are evolving to respond to changes in market demands. Computer numerically controlled (CNC) machines and computer-aided systems (CAx) are the results of the fusion of the technologies made possible by these revolutions.

In the past 50 years there have been many advances in CAx resource capabilities. Various vendors have tried to enable users to utilise the potential of CAx resources by modifying the original standard interfaces and adding proprietary extensions. This has resulted in development of a plethora of standards, programming languages and vendor specific subroutines. Consequently resources with different configurations from different vendors are not interoperable. The lack of interoperability has started to hinder the flexibility of manufacturing enterprises in terms of resource fluidity significantly: CAx resources cannot be assigned to tasks that were supposed to be completed by another resource.

The main contribution of this research to knowledge is a new vision towards interoperability leading to a novel framework that is specified and designed for enabling interoperability between CAx systems. The framework increases the availability of high-level, semantically homogenised information for the resources within the CAx chain. A prototype implementation of the interoperable CAx framework is utilised to demonstrate the capabilities of the framework. The demonstration is conducted using an industrially inspired test component where the component is programmed and machined interchangeably on two different CNC milling machines and a CAD/CAM system.

The research shows that a semantically homogenised, high-level manufacturing lexicon together with an effective abstraction of CAx resources and a powerful communication mechanism can provide a robust and comprehensive framework to realise interoperability in CAx manufacturing. The ingenuity of the proposed framework is that it not only enables adopters to utilise current CAx resources in an interoperable manner but also provides a totally versatile platform for the development of future generations of interoperable CAx systems.

List of Abbreviations

ADAM	Automated Drafting and Machining
AFNOR	Association Française de Normalisation
AIM	Application Integrated Model
ARM	Application Reference Model
BNCL	Base Numerical Control Language
CAD	Computer-Aided Design
CAE	Computer-Aided Engineering
CAM	Computer-Aided Manufacturing
CAPP	Computer-Aided Process Planning
CAV	Computer-Aided Verification
CAx	Computer-Aided System
CIM	Computer Integrated Manufacturing
CIMOSA	Computer Integrated Manufacturing Open System Architecture
CNC	Computer Numerically Controlled
CORBA	Common Object Request Broker Architecture
DNC	Distributed Numerical Control
DSDL	Document Schema Definition Language
DTD	Document Type Definition
ERP	Enterprise Resource Planning
GUI	Graphical User Interface
IDL	Interface Definition Language
IGES	Initial Graphics Exchange Specification
IP	Internet Protocol

IP^3AC	Integrated Platform for Process Planning and Control
KIF	Knowledge Interchange Format
KQML	Knowledge Query and Manipulation Language
LM	Layered Manufacturing
NC	Numerically Controlled
OMG	Object Management Group
OOT	Object Oriented Technology
PDM	Product Data Management
PERA	Purdue Enterprise Reference Architecture
PLC	Programmable Logic Circuit
PPC	Production Planning and Control
SDAI	Standard Data Access Interface
SET	Systeme d'Echange et de Transfert (Data exchange and transfer standard specification)
SOAP	Simple Object Access Protocol
STEP	Standard for the Exchange of Product data model
UDDI	Universal Description, Discovery and Integration
UML	Unified Modelling Language
URL	Uniform Resource Locator
VDA-FS	Verband der Automobilindustrie - Flächenschnittstelle (the organisation of the automotive industry - surface translation format)
VM	Virtual Manufacturing
VRML	Virtual Reality Modelling Language
WSDL	Web Services Description Language
XML	Extensible Mark-up Language

Definitions

Adaptability - The flexibility and agility of a manufacturing enterprise in handling changes in resources, jobs and strategies.

CAx - Computer aided systems utilised in a manufacturing enterprise. Includes CAD, CAM, CAV, CNC etc. In this thesis the term CAx has been used interchangeably with CAD/CAM/CNC as these are the most prevalent computer aided systems in prismatic part manufacture.

Class - A blueprint for creation of an object. An object is an *instance* of a class. The class specifies the attributes and methods associated with all objects that are instantiated from it.

Feature Based System - A CAx system that incorporates a library of pre-defined geometrical shapes or "features" to describe a product or manufacturing process

G&M Codes - The CNC machine programming language standardised in ISO6983. The term "G&M codes" has occasionally been used in this thesis to refer to all "machine axis movement description" languages.

Inheritance - A class can be the subtype of another class. In this case, the child class *inherits* all of the parent class's attributes and methods and can overwrite or amend them.

Integration - The process of incorporating parts, components, elements into a larger defined unit, set, whole.

Interface - The contact point between a class and the outside world. When a class *implements* an interface, it promises to provide the behaviour defined in that interface.

Interlingua - A language neutral representation of semantic and grammatical concepts.

Interoperability - The ability to seamlessly transfer information from one computer system to another, while maintaining the integrity of the information.

Object - A packaging of related state and behaviour in software. Object-Oriented programming utilises objects to model the real-world objects and encapsulate data and actions in them. The state is captured in an objects *attributes* and its behaviour is defined using *methods.*

Object Persistence - The ability to store an object's state to disk or another persistent information storage system to be retrieved later.

Polymorphism - A method in a class can have a number of different behaviours based on the class that is passed onto it as a parameter. This powerful feature in Object-Oriented programming is called polymorphism.

Prismatic Part - A mill machined part is defined to be prismatic when all of the planes defining the part's surface geometry are either parallel or perpendicular to the spindle axis of the milling machine used in its manufacture. In this research the term is only used for parts that can be machined using one tool axis direction (i.e. can be manufactured on a 3-axis milling machine in a single setup).

Resource Abstraction – Resource abstraction is the process of hiding a resource's complexity from the outside world and providing it with standardised input/output interfaces.

Resource Fluidity - Ability to use different available resources interchangeably with minimum effort

Semantic Interoperability - In order to obtain mutual understanding of interchanged data, the actors have to share a model of what the data represents. Semantic interoperability is achieved when such mutual understanding exists.

Semantic Interpretation - The process of discovering semantics conveyed through a specific syntax of data.

Semantics - The implied meaning of data within a specific context with respect to their role in a system.

Standardised Input / Output - Data transferred from/to a CAx resource where the semantics and syntaxes have been homogenised with those of the manufacturing Interlinuga.

Syntax - The rules that regulate the format of representation of information within a context.

List of Figures

Figure 1.1 - Organisation of thesis chapters 3
Figure 2.1 - Increasing information transfer bandwidth in the CAD/CAM/CNC chain 5
Figure 2.2 - Research boundaries within the context 8
Figure 3.1 - STEP data access interface (SDAI) overall view (Adapted from Sandakly 2001) 19
Figure 3.2 - A framework for intelligent CNC based on STEP-NC (Suh and Cheon 2002) 27
Figure 3.3 - Overall diagram for an ISO14649 compliant CNC milling machine (Lee et al. 2006) 28
Figure 3.4 - Intelligent process planning system based on STEP-NC (Amaitik and Engin Kilic 2007) 29
Figure 3.5 - Conceptual architecture of I^2NC 34
Figure 3.6 - Enterprise information integration framework (Giachetti 2004) 43
Figure 3.7 - Organisation Network (Adapted from Zaidat et al. 2005) 44
Figure 3.8 - UMEO classes and instances (Zimmermann et al. 2002) 45
Figure 3.9 - Levels of conceptual interoperability (Tolk and Muguira 2003) 46
Figure 3.10 - Criteria for determining the openness of control systems (Pritschow et al. 2001) 47
Figure 3.11 - CAM-CNC combinations for postprocessors 51
Figure 3.12 - The vision of STEP-NC replacing the postprocessors 52
Figure 4.1 - A categorization of the manufacturing information for CNC machining of prismatic parts 55
Figure 4.2 - Block diagram of the interoperable CAx framework 57
Figure 4.3 - The overall UML sequence diagram for the interoperable CAx framework 58
Figure 4.4 - Semantic interpretation and syntax conversion as enablers of interoperability 58
Figure 4.5 - Comparison of two CNC programs with different levels of information 59
Figure 4.6 - The required information models for the interoperable framework 60
Figure 4.7 - An example of manufacturing information represented using STEP-NC entities 61
Figure 4.8 - Functional overview of the interoperable CAx framework 63
Figure 4.9 - Overall view of the interoperable CAx framework 64
Figure 5.1 - Information flow in state-of-the-art CAx chain 66
Figure 5.2 - The information flow in a STEP-NC compliant CAx chain 67
Figure 5.3 - IDEF0 diagram of resource abstraction functionality 68
Figure 5.4 - CAx resource abstraction 68
Figure 5.5 - Abstracted resource interface 69
Figure 5.6 - A CNC machine with direct implementation of resource abstraction 69
Figure 5.7 - Indirect implementation of CAx resource abstraction 70
Figure 5.8 - UML diagram of the resource abstractor class 71
Figure 5.9 - The resource abstractor classes for Siemens 840D and GE Fanuc controllers 72
Figure 5.10 - A simple prismatic component 72
Figure 5.11 - STEP-NC process plan for the simple prismatic component 73
Figure 5.12 - Output generated by the proprietary output method of the Siemens 840D abstractor object 73
Figure 5.13 - Output generated by the proprietary output method of the GE Fanuc abstractor object 74
Figure 5.14 - XML definition of Siemens 840D controller on a Bridgeport machine 75
Figure 5.15 - XML definition of GE Fanuc controller on a Wadkin machine 76
Figure 5.16 - Resource abstraction for a simple process plan 77
Figure 6.1 - Information layers required for CNC manufacturing 79
Figure 6.2 - Object oriented encapsulation of manufacturing information in IP^3AC 82
Figure 6.3 - An example of multiple inheritances in ISO14649 83
Figure 6.4 - Defining EXPRESS's multiple inheritances using Java interfaces 83
Figure 6.5 - Generic input and output methods for IP^3AC classes 86
Figure 6.6 - IDEF0 representation of the EXPRESS Translator 89
Figure 6.7 - IDEF0 representation of activity A3 - Java syntax generation 90
Figure 6.8 - Storage and retrieval of the STEP-NC Program Structure 91
Figure 6.9 - IP^3AC STEP-NC Tree Viewer 92

Figure 6.10 - STEP-NC 3D part viewer 93
Figure 7.1 - Various approaches for information transfer from one CAx resource to another CAx resource... 96
Figure 7.2 - An overview of the Aglets mobile agent environment 97
Figure 7.3 - UML class diagram of a G&M code text file data source 98
Figure 7.4 - UML class diagram of the carrier agent 98
Figure 7.5 - Overview of the monitor agent. 99
Figure 7.6 - An example of agent interactions in a global manufacturing enterprise. 100
Figure 7.7 - Aglet proxy used to send and receive messages to aglets 100
Figure 7.8 - Overall view of the abstraction agent 101
Figure 7.9 - UML sequence diagram showing the initial agent registration sequence 102
Figure 7.10 - UML sequence diagram showing information transfer phase 1 - abstraction agent migration 103
Figure 7.12 - Overall view of the mobile agent based information transfer framework 106
Figure 7.13 - UML Class diagram of the agent based information transfer framework 106
Figure 8.1 - Interoperable CAx framework components implemented in the prototype 110
Figure 8.2 - UML Class diagram for the *express_entity* 111
Figure 8.3 - The IP^{3}AC input class 112
Figure 8.4 - UML class diagram for the output class 112
Figure 8.5 - The interoperable CAD/CAM system user interface 113
Figure 8.6 - The hole creation interface in the interoperable CAD/CAM system 114
Figure 8.7 - STEP-NC generated in the interoperable CAD/CAM system 115
Figure 8.8 - Overall functional view of the Interoperable CAD/CAM system 116
Figure 8.9 - ShopMill user interface 116
Figure 8.10 - Overall view of the ShopMill Abstractor 118
Figure 8.11- The Heidenhain iTNC530 smarT.NC programming interface 119
Figure 8.12 - Heidenhain proprietary G&M Code cycles 119
Figure 8.13 - The overall view of the Heidenhain iTNC530 CNC resource abstractor 120
Figure 8.14 - Information transfer overview in the interoperable framework prototype 121
Figure 8.15 - MPF abstraction object invoked through the interoperable CAD/CAM system 121
Figure 8.16 - Key classes of the interoperable framework prototype 122
Figure 9.1 - The manufacturing scenario for evaluation of the prototype 123
Figure 9.2 - Test component drawing 124
Figure 9.3 - Planar facing operation 125
Figure 9.4 - Outermost pocket machining with boss 125
Figure 9.5 - Outer pocket machining with boss 126
Figure 9.6 - Inner pocket machining 126
Figure 9.7 - Second tier pocket 1 machining 127
Figure 9.8 - Second tier pocket 2 machining 127
Figure 9.9 - Circular pockets machining 128
Figure 9.10 - Rectangular pocket machining 128
Figure 9.11 - 8mm holes drilling 129
Figure 9.12 - 3mm holes 129
Figure 9.13 - ShopMill code excerpt 130
Figure 9.14 - Two dimensional ShopMill simulation 131
Figure 9.15 - Three dimensional ShopMill simulation 131
Figure 9.16 - Machined part 132
Figure 9.17 - The test component in the interoperable CAD/CAM system 133
Figure 9.18 - Heidenhain cycles 134
Figure 9.19 - Heidenhain simulation of the test part 134
Figure 9.20 - Manufacturing scenario using state-of-the-art 135
Figure 9.21 - Manufacturing scenario using the interoperable CAx framework prototype 135
Figure 10.1 - Paradigm shift from CAx chain to interoperable CAx network 141
Figure 11.1 - Extension of the framework to integrate ongoing STEP-NC research 147

Figure 11.2 - Utilising the research to realise the Manufuture vision for interoperability 148
Figure 11.3 - An overview of the universal manufacturing platform 150

List of Tables

Table 3.1 - A list of STEP parts within the scope of research 16
Table 3.2 - Entity definition in EXPRESS 17
Table 3.3 - ISO10303-21 Excerpt 17
Table 3.4 - Research on STEP-NC as an integrator 26
Table 3.5 - STEP-NC research with focus on milling technology 30
Table 3.6 - Turning and Turn/Mill STEP-NC research 32
Table 3.7 - STEP-NC research with focus on technology other than milling and turning 35
Table 3.8 - STEP-NC reviews 36
Table 3.9 - Software agent typology proposed by Nwana (1996) 37
Table 3.10 - The effects of wide adoption of STEP-NC in manufacturing 50
Table 6.1 - Key entities in ISO14649 and their respective information layer 81
Table 7.1 - Fundamental agent messages in the mobile agent based information transfer framework 107
Table 8.1- Methods defined in the *express_entity* class 111
Table 8.2 - Methods provided by the output class 112
Table 8.3 - ShopMill features and equivalent STEP-NC entities 117
Table 10.1 - Comparison of the various interoperability approaches 142

1. Introduction

Many modern manufacturing paradigms have their roots in the days of the industrial revolution. Concepts such as mass production and standardisation have been the key enablers for making manufacturing an indispensable part of the modern human civilisation. The advent of the information revolution and the unfolding of the age of communications are now making fundamental changes in our society. For manufacturing to remain competitive in the global market, it is essential for practitioners to review, reform and where necessary, reinvent the traditional concepts of manufacturing according to these changes. The manufacturing economy of yesteryear is giving way to a knowledge based economy that values individual tastes and personalisation as well as minimisation of lead times and lowering production costs.

Numerically Controlled (NC) and Computer Numerically Controlled (CNC) manufacturing devices have been developed and are continuously being enhanced as tools to meet these evolving demands. The accuracy, repeatability and performance of these machine tools can be a critical enabler of mass personalisation in realising knowledge based engineering in the domain of manufacturing.

CNC machine tools, in particular, have been developed to automate metal cutting processes. With each new generation of CNC machine tools, the hardware capabilities have been significantly augmented. The controllers have evolved from simple memory-less electronic devices to computers with substantial memory and processing power to effectively control this enhanced machining hardware. This added complexity has made programming CNC machines more difficult. Thus, the arduous task of programming, itself, has been automated through the use of Computer Aided Design (CAD) and Computer Aided Manufacturing (CAM) software packages that generate suitable part programmes to control the CNC machines. The unsynchronised development of hardware and software however, has resulted in the waste of significant computing potential on the controllers. Currently, extremely capable machines are programmed in a manner that is fundamentally identical to that of the pioneering NC machines of the '60s and '70s.

CNC makers have tried to solve this problem independently by adding the required software functionality to their controllers through proprietary additions. At the same time the diverse, and sometimes hasty, developments in personal computer technology, the current platform of choice for CAD/CAM systems, have created an intricate array of alternative hardware and software combinations for manufacturing companies. A significant number of these machine, controller and CAD/CAM combinations, however, do not provide interoperability with each other.

Interoperability or the ability to seamlessly transfer information from one computer system to another without loss of data leads to resource fluidity. Resource fluidity is a key factor for adaptability of an enterprise. In today's economy with the globally marketed customer-tailored-products where demand is volatile, adaptability is critical for survival of a company. The current CAD/CAM/CNC systems fall short of delivering this requirement and therefore a major paradigm shift towards a more adaptable framework has become necessary.

In this research a novel vision for this paradigm shift has been outlined and an interoperable framework comprised of CAx systems and CNC machines has been realised to support and enable this leap.

The organisation of the research is such that first the aims, objectives and the scope have been presented. A review of the existing literature on manufacturing integration and interoperability follows. The research gaps and opportunities have then been identified. In the theoretical phase of the work, the novel framework for realising CAx interoperability has been specified, envisioned and developed. The framework consists of a number of elements that have then been presented in full detail. In the experimental phase, a prototype implementation of the interoperability framework has been realised, demonstrated and evaluated. A number of topics of discussion raised in the course of the research then follow. Finally the conclusions drawn in the course of the research together with areas with potential for future research have been presented. Figure 1.1 shows the organisation of the different chapters and their contents within the context of the research.

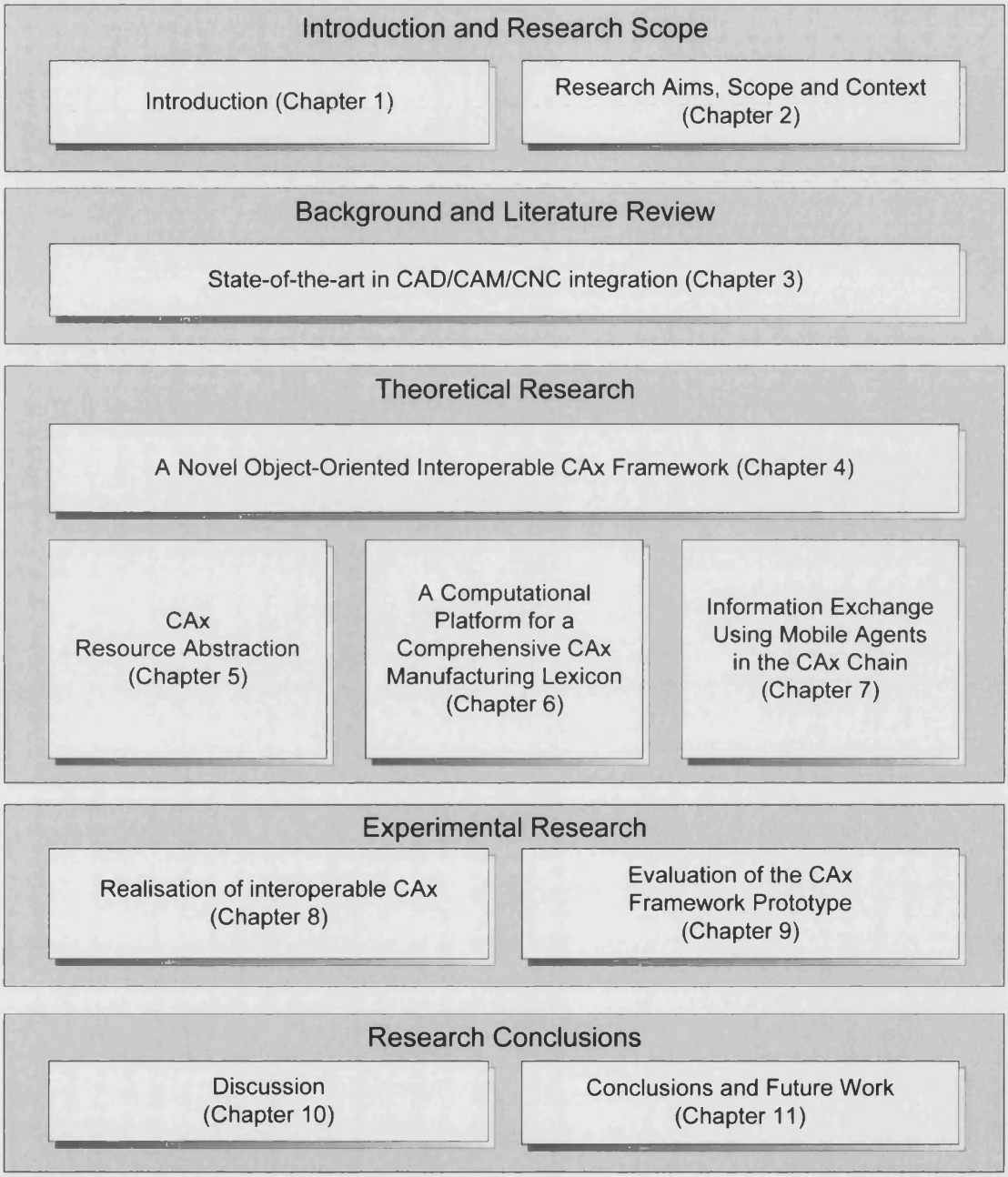


Figure 1.1 - Organisation of thesis chapters

2. Research Aims, Scope and Context

2.1. Introduction

This chapter outlines the aims, scope and boundaries of the research. To do this, the boundaries of an interoperable CAx system will be established and the aims in enabling interoperability in the system are identified. The scope of the research specifies the particular perspective of the author in determining the relevant issues pertaining to the development of interoperability and positions the work in relation to other research on interoperability. As a result the overall context of the research is ascertained.

2.2. Research Aims

Manufacturing enterprises have access to a wide range of hardware and software tools in the CAD/CAM/CNC domain. Multi-axis machining centres capable of high-speed machining of complex geometries utilising a number of manufacturing processes are examples of the latest generation of hardware. Feature based CAD systems with realistic 3D visualisation of the models and CAM systems capable of machine toolpaths optimisation and verification are software examples. These tools form the state-of-the-art CAD/CAM/CNC chain and are individually powerful and use the latest available technologies. The diverse, and occasionally hasty, advances in the various technologies involved in their development however, have resulted in a plethora of standards and protocols; most of them proprietary and vendor specific.

Due to this multi-standard environment, the integration of CAD/CAM/CNC has been slow and interoperability between the various components is almost nonexistent. Currently the standards employed in practice, utilise the lowest common denominator of the systems exchanging information. As a result, the only means available today for sending an optimised process plan from a CAM system to a CNC machine is to use G&M codes, a standard formalised 25 years ago (ISO 1982). This low-level standard has become a bottleneck and together with uni-directional data transfers constitutes a major hindrance in achieving interoperability and adaptability for manufacturing enterprises employing CNC technology.

The main hypothesis of this research is that it is possible to move towards interoperability and adaptability in CNC manufacturing by adopting the interoperable framework proposed in chapter 4.

The key enabler for this, is information availability and semantic interpretation of manufacturing concepts. Information availability can be increased by replacing the current low-level information exchanges with high-level information links between various components of a CAD/CAM/CNC chain. This change is depicted in Figure 2.1.

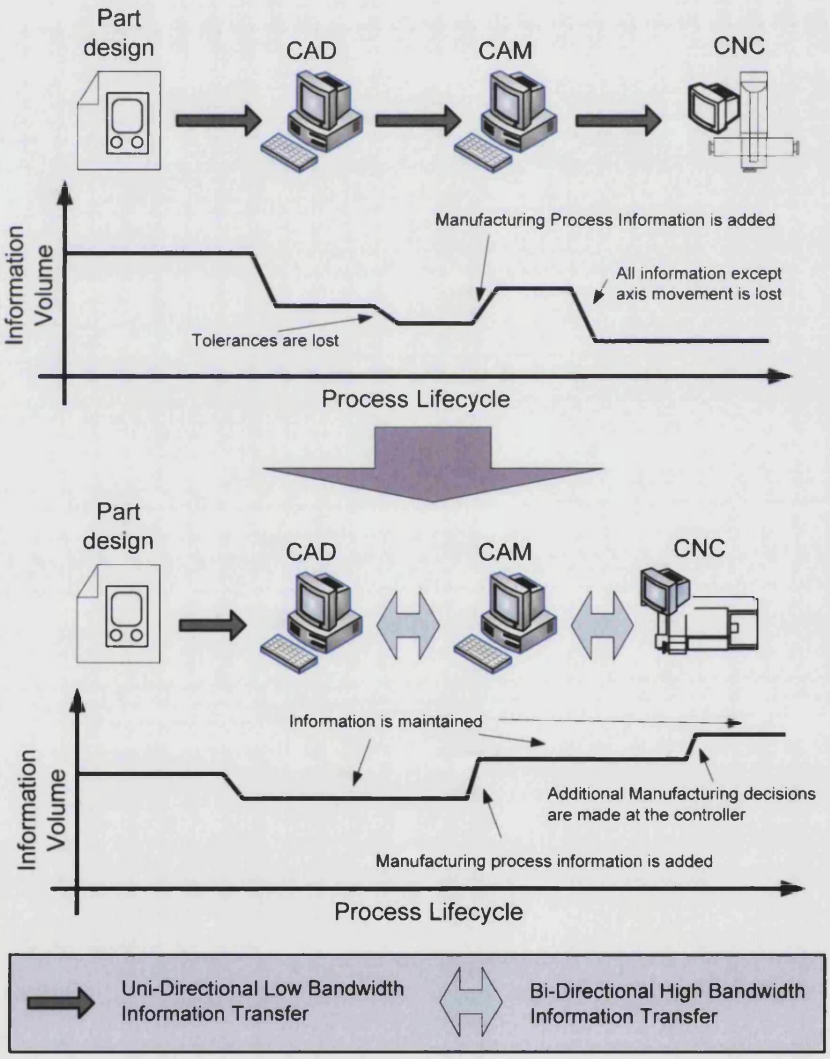


Figure 2.1 - Increasing information transfer bandwidth in the CAD/CAM/CNC chain

Semantic interpretation allows the heterogeneous manufacturing information stored using various syntaxes across CAx systems to be homogenised and expressed using a single representation.

The aim of this research is to conceptualise, design, implement and realise an interoperable framework based on high-bandwidth information transfer and semantic homogenisation to prove the hypothesis. This system will allow high-level integration and interoperability of the existing computer software and hardware systems in CNC manufacturing with state-of-the-art CNC controllers.

2.3. Research Methodology and Objectives

To achieve the aims outlined above, a deductive approach has been chosen as the research methodology. In this approach a theory is established and then the validity of the theory is proven through observations and practical findings (Bryman, 2004). As the research problem is clearly defined as the lack of interoperability in the current generation of CAD/CAM/CNC systems, the research can be considered as being constructive (Lukka, 2003). This type of research is undertaken after exploratory research where the problems within a specific domain are defined. The results derived from the research can later be used to conduct empirical research. A novel computational framework is the main construct in the research and its effectiveness is demonstrated utilising a test component on a prototype implementation of the framework.

To realise this research methodology, a number of objectives have been defined:

- Reviewing the state-of-the-art of CNC manufacturing in regards to integration and interoperability to formulate the interoperability problem within the boundaries and the scope of the research

- Specifying and designing a novel framework for realisation of interoperability in the CAx chain. The framework will increase the availability of information throughout the CAD/CAM/CNC chain and introduce methods to maintain information integrity. This objective can be achieved by:

- Creating an effective information coding mechanism according to a unified manufacturing information representation language.
- Abstraction of CAx resources to enable information exchange through generic and standardised interfaces.
- Identifying secure, robust and reliable means of communications between CAx resources.
- Realising a prototype interoperable CAx chain based on the requirements

- Evaluating the prototype interoperable CAx chain by using an industrially inspired test component and state-of-the-art commercial software and hardware tools (i.e. CNC machines and CAD/CAM systems).

2.4. Research Context and Boundaries

The context of this research is CAD/CAM/CNC within manufacturing enterprises. A number of boundaries have been identified within the context to allow the research to focus on the key issues of information availability and semantic interpretations. These boundaries are illustrated in figure 2.2 and are defined as follows:

2.4.1. Interoperability and Adaptability in Manufacturing Integration

Integrating manufacturing systems is a wide topic and encompasses a large body of research. The research ranges from integration of embedded systems within machinery (Bunse and Gross, 2006) to integration of process monitoring systems within chemical plants. In this research, the focus is specifically on interoperability and adaptability in CNC manufacturing.

Interoperability is defined as the capability of different computer systems to exchange information seamlessly without manual intervention (Ray and Jones, 2006). Adaptability is defined as the flexibility and agility of a manufacturing enterprise in handling changes in resources, jobs and strategies (Kosonen and Doz 2006). Interoperability is a relatively new topic of research. As a result it has been necessary to consider other views of integration in the manufacturing domain in the development of the framework.

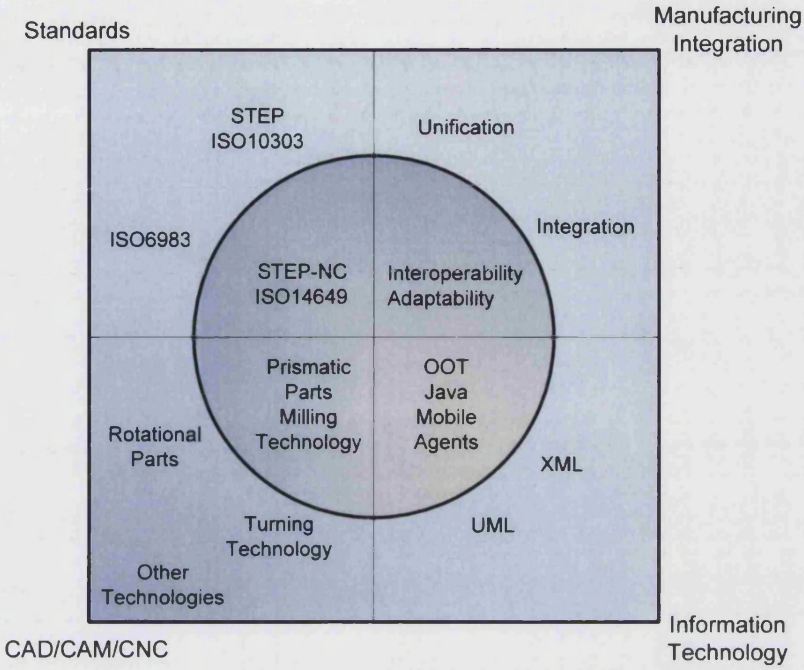


Figure 2.2 - Research boundaries within the context

2.4.2. Prismatic Parts and Milling Technology in CAx

CAD/CAM/CNC technology is now used to manufacture various types of components using a number of processes. From 3-axis milling used to machine prismatic parts to 5-axis turn/mill for asymmetric rotational parts to wire-EDM to rapid manufacturing (Xu and Newman 2006).

To focus on the interoperability challenges and avoid the complexity of dealing with multiple manufacturing processes, prismatic parts and 3-axis milling, have been chosen as the main area of interest within the CAD/CAM/CNC domain.

2.4.3. STEP-NC (ISO14649) in Standards

Today a vast number of standards and proprietary information models are being used to describe manufacturing process, products and resources. ISO14649 (ISO 2003a), a standard initiated in the mid '90s and still under development has been proposed to facilitate the integration of CNC manufacturing systems.

This standard has been chosen as the basis of this research as it provides suitable hierarchical data structures to store manufacturing information for prismatic components and process

plans. The standard is used extensively in chapter 6 in creating a comprehensive manufacturing lexicon. Other standards such as ISO10303 (ISO 1994) and ISO6983 (ISO 1982) were referred to where ISO14649 did not provide the necessary constructs.

2.4.4. Application of Information Technology

To design and implement the interoperable CAx framework, a number of information technology tools have been utilised. Object oriented technology has been chosen together with Java as the programming language as they provide a robust and portable programming environment. Mobile agents have been investigated in chapter 7 as communication providers for high bandwidth information exchange across the chain of resources.

2.5. The Scope of the Research

To achieve the objectives defined in 2.2 the following sections are identified as the research scope:

2.5.1. Review of the State-of-the-Art in CAD/CAM/CNC Integration

Different approaches and various tools have been used by researchers to tackle integration problems within the CAD/CAM/CNC domain. The existing research has been reviewed and assessed in chapter 3 to identify potential methods for achieving interoperability. This literature review has been utilised to identify the research gaps that form the foundations of the specification requirements for the current research.

2.5.2. Specification of a Novel Framework for Realisation of Interoperability

Based on 2.5.1 a theoretical framework relying on standards and information technology has been developed in chapter 4 to establish the requirements of interoperable CNC manufacturing. Methodologies such as IDEF-0 and the Unified Modelling Language (UML) have been utilised to represent the necessary processes and information models.

2.5.3. Defining Manufacturing Resource Abstraction

The information flow within the interoperable framework has been analysed to identify the data coupling points. These points represent information transformation bottlenecks that indicate a change in the semantics of the data as opposed to syntax modifications. An

abstraction process for homogenising the semantics and standardisation of the syntaxes is then proposed in chapter 5.

2.5.4. Investigation of Utilising a Comprehensive Manufacturing Lexicon

A computational vehicle to realise a comprehensive and inclusive manufacturing lexicon has been proposed, defined and investigated in chapter 6 as a structured method for encoding manufacturing information. This computational tool, called the integrated platform for process planning and control (IP^3AC) forms the basis for implementation of the interoperable framework.

2.5.5. Exploration of Mobile Agents as Information Carriers

In order to implement the interoperable framework a data communication infrastructure is required to ensure the integrity of information transfer among CAx resources. Mobile agents have been explored in chapter 7 to identify the feasibility of their application as information carriers in an interoperable CAD/CAM/CNC chain.

2.5.6. Realisation of a Prototype Interoperable CAx Framework Implementation

The information encoding developed in chapter 6 has been used in conjunction with the abstraction process from chapter 5 to implement a prototype interoperable CNC manufacturing system in chapter 8.

2.5.7. Evaluation of the Interoperable CAx Framework Prototype

The system has been evaluated using a prismatic test component based on those utilised in the industry in chapter 9. The level of interoperability provided by the framework has been assessed by manufacturing an industrially oriented test component on different CNC machines with different controllers.

3. State-of-the-Art in CAD/CAM/CNC Integration

3.1. Introduction

In this chapter the state-of-the-art solutions to manufacturing interoperability problems and CAD/CAM/CNC integration have been presented. Various philosophies and techniques that are used to transfer information from one manufacturing system to another are investigated. A review of the existing literature for each method is then provided with a critique to highlight the advantages and disadvantages of each approach. This data is then used to identify the research gaps that exist in the current solutions. Additionally a list of the author's publications in the research area can be found in appendix A.

3.2. The CAx Chain: A Short History and Background

Since the industrial revolution machining metals to create specific shapes became important in the production business. John Wilkinsion invented the first metalworking lathe in England in 1775 which was used to bore cannons (Agapiou, 2006). The machine was later adapted to bore steam engines. This process at the time, was the only one capable of supporting the tolerances that was required by steam engines.

Milling machines were invented later by Eli Whitney in 1818 (Meyers and Slattery, 2001) to allow machining parts that were not necessarily circular in shape. These milling machines first found use in making rifles for the US government.

As the precision requirements in the parts increased over the years, repeatability of manufacturing processes became essential due to the needs of the mass production economy. Thus more accurate methods for controlling the movements of axes in metal-cutting machines were sought.

In an effort to meet this need John Parsons and Frank Stulen pioneered numerical control (NC) for machine tools in the 1940s (Kochan, 1985). They worked in collaboration with IBM and MIT and when in 1952 numerically controlled servomechanisms were invented in MIT the first "true" NC machine was realised (Dorf and Kusiak, 1994). In 1958 John Parsons received

a patent for "a motor controlled apparatus for positioning machine tool" (Parsons and Stulen, 1958).

Computer numerically controlled (CNC) metal cutting machines were introduced in the 1970s with the introduction of Minicomputers (Talavage, 1988) and replaced the hardwired logic of the older NC machines with a programmable interface that could be coded to produce different parts.

The programming language used to program these machines was an extension of the codes Gerber Scientific Instruments used for their two axis scientific photoplotters (Schroeder 1998). The two dimensional language of the plotters was extended to include a third (Z) axis and with the addition of special switching instructions was capable of supporting three axis milling machines. This language became known as G-Codes for Gerber Scientific.

A file containing G-Codes would comprise many lines of text that would be interpreted as moving instructions for the servomechanisms connected to various axes of the machines (Smid, 2003). The language was later expanded to cover lathes and later on, the 4^{th} and 5^{th} axes on milling machines (Radhakrishnan et al. 2000). G-Codes became the de-facto standard for programming and were formalised in RS274D, ISO6983 and DIN66025 (Liu et al. 2007).

CNC machines made programming the metal removal process much easier. They provided engineers with a programming language that could be used to develop programs stored in files. The files were read in the machine first through punched cards and later through magnetic tapes and with the advent of DNC (Lynch 1992) directly from a computer through RS232 interfaces.

For two decades upto the 1970 program files had to be created manually. An engineer needed to go through product design and determine the path that the tool needed to move on (toolpath) to produce the desired part. They would then, analyse the tool path and interpret it as axis interpolations that could be programmed using G-codes (Mattson 2001).

In the 1970s CAD and CAM systems became commercially available. The earlier systems from the 1960s like SKETCHPAD (Sutherland 1963, Sutherland 2003) matured into more

advanced systems (Grabowski, 2001:196). A CAD system allowed engineers to enter part drawings into the computer to create a digital representation of the part geometry. A CAM system would then utilise the process plan entered using specific commands by the engineer to generate the G-Code files automatically. This process simplified the generation of machine code for production of metal parts and reduced the required time and resources tremendously. CAD, CAM and CNC systems which now formed a computer integrated manufacturing (CIM) platform (Doumeingts 1995), became known as computer aided x (CAx) systems. CAx systems formed a chain from design to manufacture and the definition was later expanded to include systems such as Computer Aided Engineering (CAE), Computer Aided Testing(CAT), Computer Aided Plant Layout (CAPL), Product Data Management (PDM), etc. (Werner Dankworth et al. 2004).

The new market for CAD and CAM solutions became very competitive in the 1980s and 1990s. In the meanwhile numerous advances were made on the CNCs such as Multi-Axis machining centres and open-system controllers (Xu and Newman, 2006). Every software vendor tried to support the new abilities of the latest generation of machines and offer users new features to increase their market share. Due to market competition, both the CAD/CAM vendors and the CNC manufacturers were very protective of the advances they were making and therefore started utilising proprietary standards for the enhancements that they introduced in their products.

For example, CNC controller manufacturer's added non-standard G codes to the ISO6983 standard to support the additional axes on the machining centres resulting in various dialects for different machine controller combinations (Proctor et. al 2002). As another example, CAD vendors utilised proprietary formats like DWG (Grabowski 2001:169) and DXF (Autodesk 2007) to store geometry information. Companies like Mazak (Polywka and Gabrel, 1992:342) introduced total solutions where both the hardware and the software systems throughout the CAx chain were provided by the vendor. While each system could be seamlessly integrated with other systems in the CAx chain from that company, it would not be possible to use another company's solution.

Gradually the advances in the CAx technology in the past 35 years resulted in a plethora of languages and proprietary standards. As a result it has become almost impossible to replace a component of a CAx chain with one from another vendor without considerable effort (Xu and He 2004). Furthermore, the CAM systems also needed post-processors for each machine-controller configuration to be able to generate the correct dialect of G-codes for that specific combination (Hardwick and Loffredo 2006).

Consequently the CAx chain has become a non-interoperable set of isolated islands of automation where information links have to be maintained through low-bandwidth information transfers suitable for the lowest common denominator (i.e. a solid 3D CAD system is used together with a CAM system capable of handling sculptured surfaces but only axis movement information is retained when passed on to the CNC machine for manufacturing).

3.3. Integration Standards

A number of existing standards can be utilised to improve CAx system interoperability. These include standards for information exchange and standards for supporting infrastructures. The information exchange standards utilised in manufacturing are outlined below and include IGES, VDA-FS, SET, STEP, STEP-NC, KIF and XML. The supporting infrastructures are also investigated in detail below and include CORBA, Web Services and KQML.

3.3.1. IGES, VDA-FS, SET

The development of standards for information exchange started with The Initial Graphics Exchange Specifiation (IGES) formally known as Digital Representation for Communication of Product Definition Data (Parks 1984). This standard was developed as the result of the IGES project started in 1979 by a group of CAD users and vendors including Boeing, General Electric, Xerox and Computervision with the support of the National Bureau of Standards (now known as NIST). IGES was a standard format that proposed to complement proprietary standards of the various CAD systems to allow them to exchange information seamlessly.

In 1988 the US department of defence made it a requirement for all contractors to deliver electronic drawings of their designed systems using IGES (DOD 1992). This resulted in wide adoption of the standard by vendors.

Due to shortcomings in IGES for specific industries a number of other standards were developed: Verband der Automobilindustrie - Flächenschnittstelle (VDA-FS) was developed by the German organisation VDA to support the automotive industry (Strasser 1999). This standard is restricted to free-form surface data transformation, one of the limitations in IGES. The French Systeme d'Echange et de Transfert (SET) is an AFNOR standard similar to IGES but with more compact data structures (AFNOR 1989). Neither of these standards became as commercially viable as IGES.

3.3.2. The Standard for the Exchange of Product Data Model (STEP)

In 1984 in an attempt to standardise the exchange of product information throughout the entire product lifecycle, development started on a new ISO standard called STEP (Eastman 1999). The official number for the standard is ISO 10303.

STEP aims to provide a standard that can be utilised to exchange data between various CAx systems. It is comprised of various parts currently at different stages of development where specific parts of the standard are referred to as "ISO10303-partnumber". The first parts to be completed were CAD information exchange standards namely ISO10303-203 and ISO10303-204. This was followed by the development of manufacturing information exchange protocols. Table 3.1 shows the list of the parts of the standard that are related to the context of this research.

(i) ISO10303-11: EXPRESS language

The EXPRESS language (ISO 1994a) is used as the main description method for the data models within the STEP framework. EXPRESS is a data modelling language that utilises Object Oriented like concepts to allow modelling of domains within the field of product data. The application protocols (APs) models are defined using EXPRESS schemas.

The fundamental constructs in EXPRESS are entities and types. Table 3.2 shows the definition of an entity with EXPRESS. Entities are more complex than types as they allow subtype/supertype relations and instances of these have identifiers unlike instances of types. Primitive types like Strings, Integers and Floating Point Numbers are therefore defined as types in EXPRESS. Enumerations with instances assuming one of the values in a set of values are also defined as types, i.e. an enumeration of rotation direction from the set of clockwise and counter clockwise.

Table 3.1 - A list of STEP parts within the scope of research

Description methods	
Part 11	EXPRESS language reference manual
Implementation methods	
Part 21	Clear text encoding of the exchange structure
Part 22	Standard data access interface specification
Part 23	C++ language binding of the standard data access interface
Part 24	C language binding of the standard data access interface
Part 27	Java TM programming language binding to the standard data access interface with Internet/Intranet extensions
Part 28	XML representation for EXPRESS-driven data
Application protocols (APs)	
Part 203	Configuration controlled design.
Part 204	Mechanical design using boundary representation
Part 214	Core data for automotive mechanical design processes
Part 224	Mechanical product definition for process plans using machining features
Part 238	Application interpreted model for computer numeric controllers

Currently STEP is not capable of representing functionality in the form of methods in the model schema. Consequently an important notion is that an EXPRESS schema describes a valid population in the universe of discourse and does not specify how the population can be

transformed into a valid population within another EXPRESS schema. It is therefore possible to use an EXPRESS schema to define a valid data model for the CAD definition of a component, but not the transformations required to generate a CAM model based on that CAD model.

The EXPRESS schema however supports relations between various entities in different domains. It is therefore possible to link a specific entity in one schema to another in a different schema.

Table 3.2 - Entity definition in EXPRESS

```
ENTITY drilling_type_operation  (* m0 *)
ABSTRACT SUPERTYPE OF (ONEOF(drilling_operation, boring_operation,
back_boring, tapping, thread_drilling))
SUBTYPE OF (milling_machining_operation);
cutting_depth: OPTIONAL length_measure;
previous_diameter: OPTIONAL length_measure;
dwell_time_bottom: OPTIONAL time_measure;
feed_on_retract: OPTIONAL positive_ratio_measure;
its_machining_strategy: OPTIONAL drilling_type_strategy;
END_ENTITY;
```

(ii) ISO10303-21: Clear Text Encoding of the Exchange Structure

Part 21 (ISO 1994b) of STEP is the implementation method that describes how a valid population of a specific domain within the standard can be presented using an ASCII file. The file starts with a header section followed by a data section. The header contains information about the creation of the file: names of the creators and dates of modifications. The data section contains the instances of the entities representing the population. Each entity instance is preceded by a hash number that is used to refer to that instance where needed. Table 3.3 shows an excerpt of a part 21 file.

Table 3.3 - ISO10303-21 Excerpt

```
DATA;
#1=PROJECT('SAMPLE PROJECT',#2,(#3),$,$,$);
#2=WORKPLAN('MAIN WORKPLAN',(#6,#7,#8,#9,#10,#11,#12,#13,#14,#15),$,$,$);
#3=WORKPIECE('SIMPLE WORKPIECE',#4,0.010,$,$,#52,());
#4=MATERIAL('ST-50','STEEL',(#5));
#5=PROPERTY_PARAMETER('E=200000N/M2');
#6=MACHINING_WORKINGSTEP('WS ROUGH POCKET 1',#53,#16,#21,$);
#7=MACHINING_WORKINGSTEP('WS FINISH POCKET 1',#53,#16,#22,$);
#8=MACHINING_WORKINGSTEP('WS ROUGH POCKET 2',#53,#17,#23,$);
#9=MACHINING_WORKINGSTEP('WS FINISH POCKET 2',#53,#17,#24,$);
#10=MACHINING_WORKINGSTEP('WS DRILL HOLE 1',#53,#18,#25,$);
```

```
#11=MACHINING_WORKINGSTEP('WS REAM HOLE 1',#53,#18,#26,$);
#12=MACHINING_WORKINGSTEP('WS DRILL HOLE 2',#53,#19,#27,$);
#13=MACHINING_WORKINGSTEP('WS REAM HOLE 2',#53,#19,#28,$);
#14=MACHINING_WORKINGSTEP('WS DRILL HOLE 3',#53,#20,#29,$);
#15=MACHINING_WORKINGSTEP('WS REAM HOLE 3',#53,#20,#30,$);
#16=CLOSED_POCKET('POCKET 1',#3,(#21,#22),#60,#54,(),$,#36,$,$,#37);
#17=CLOSED_POCKET('POCKET 2',#3,(#23,#24),#62,#55,(),$,#36,$,$,#38);
#18=ROUND_HOLE('HOLE 1',#3,(#25,#26),#64,#56,#31,$,#35);
#19=ROUND_HOLE('HOLE 2',#3,(#27,#28),#66,#57,#32,$,#35);
#20=ROUND_HOLE('HOLE 3',#3,(#29,#30),#68,#58,#33,$,#35);
```

(iii) ISO10303-22: Standard Data Access Interface (SDAI) Specification

SDAI (ISO 1998) defines an abstract Application Programming Interface (API) to manipulate application data using a programming language. SDAI is defined independent of any particular language however various bindings for a number of languages exist: ISO10303-23: Bindings for C++ (ISO2000), ISO10303-24: Bindings for C (ISO2001) and ISO10303-27: Bindings for Java (ISO2000a). The original plans included bindings for FORTRAN and CORBA but were cancelled.

SDAI is comprised of various components: A dictionary schema that contains a meta-EXPRESS schema to describe EXPRESS schemas, a session object to control the environment, a repository object that contains the models and instances and operations to create, manage and validate application data according to constraints and rules specified in EXPRESS. These elements are shown in Figure 3.1.

SDAI supports two methods of binding the application data to the EXPRESS data model, namely: early binding and late binding. In early binding, a specific programming language data structure is constructed for each entity in the EXPRESS model. The data structures for early binding are often automatically generated by an EXPRESS compiler. It should be noted that data structures created in a programming language will have to comply with the syntax of that language. As a result, with early binding, it would be difficult to represent EXPRESS models that are inherently incompatible with a programming language's syntax.

In late binding, an EXPRESS data dictionary is used directly to access the values of entity attributes. Queries against the data dictionary are required to retrieve and store values in the data model.

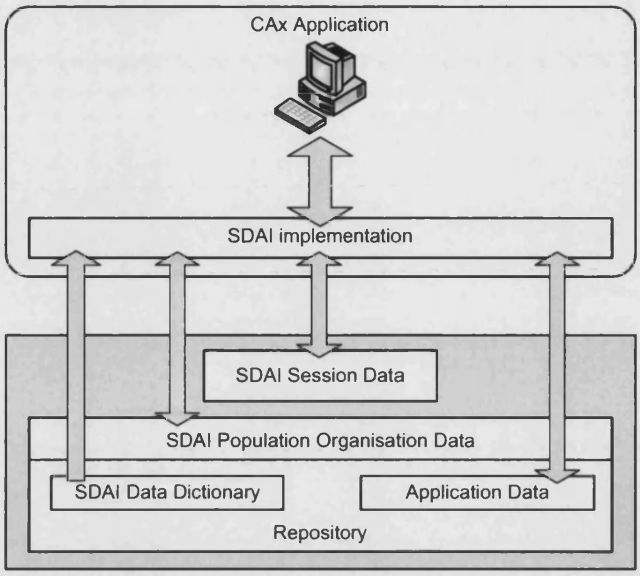


Figure 3.1 - STEP data access interface (SDAI) overall view (Adapted from Sandakly 2001)

SDAI has been utilised extensively for different protocols of STEP: Zarli and Amar (1997) developed the framework for virtual enterprise computer-based infrastructure interoperability, Eastman and Augenbroe (1998) studied its usefulness in product modelling, Hardwick et. al (2000) considered its application as the protocol to transport product data in a virtual enterprise, Zhang et. al (2000) developed the internet based STEP data exchange framework based on SDAI and Gao et. al (2003) utilised the interface in the application of product data management technologies for enterprise integration.

While SDAI is a generic language-independent specification, researchers trying to use its bindings for implementation of object-oriented applications have identified issues that make the current implementation of SDAI less than perfect for this purpose. Considering that the programming logic in Object Oriented languages is significantly different to that of procedural languages and having in mind that the original specifications of SDAI were developed with procedural languages such as C in mind, difficulties in seamless integration of the interface within object-oriented frameworks can be expected. Goh et. al. (1994) identified such difficulties in employing SDAI on top of an existing object based data access interface and Rando and McCabe (1994) identified the issues in implementing the C++ bindings.

The major software vendors that sell implementations of SDAI include: PDTec (www.pdtec.de) with ECCO Toolkit and Instance Explorer, LKSoftWare GmbH (www.jsdai.net) with JSDAI and STEP-Tools (www.steptools.com) with ST-Developer.

(iv) ISO10303-28: XML representation for EXPRESS-driven data

This part of the standard is officially known as STEP-XML (ISO 2003). Similar to ISO10303-21 this implementation method specifies the representation of a valid population of STEP entities. Instead of encoding the instances in a text file however, in part 28 the information is captured using the Extensible Markup Language (XML) (http://www.w3.org/XML/).

The current modular implementations of STEP-XML require each instance of the entities to be stored in a separate file. The possible complications that might arise from having a large number of files for a complex product has not stopped CAx researchers from implementing their integration solutions based on STEP-XML (Borselino et al. 2004, Gang et al. 2006 and Lubell 2000)

(v) ISO10303-238: Application interpreted model for computer numeric controllers

AP238 was designed as the application protocol for using STEP in exchanging information in the CNC domain (ISO2007). Each application protocol within STEP is comprised of two sections: an application interpreted model (AIM) that identifies the entities in other schemas that are related to those being defined in the application protocol using EXPRESS relationships and the application reference model (ARM) that defines the information requirements and constraints of a specific application context. While the AIM of AP238 is defined within ISO10303-238, the ARM is actually another ISO standard, namely ISO14649 better known as STEP-NC.

(vi) Other protocols in STEP

The other protocols including AP203 (ISO 2005b), AP204 (ISO 2002), AP214 (ISO 2003b) and AP224 (ISO 2006) are mainly utilised to describe part geometry. They are referenced within the other protocols of STEP such as AP238 and serve as the basis for various

representations of a part geometry ranging from vertices and lines in AP203 to manufacturing features in AP224.

3.3.3. STEP-NC

STEP-NC or ISO14649 is the ARM associated with ISO10303-AP238. Even though this standard is not a part of ISO10303, STEP description methods and implementation methods have been used to define it. ISO14649 is defined in an EXPRESS schema that contains entities and types but no relationships as those are covered in the AIM and are part of ISO10303-AP238.

ISO14649 has been developed since the late '90s to support exchange of product information on a process planning level. A valid population within the domain of ISO14649 contains a process plan specified through a logical procession of manufacturing, probing and decision activities packaged in "workingsteps". ISO 14649 is comprised of various parts each dealing with a specific CNC technology. The prominent parts are:

(i) ISO14649-1: Overview and fundamentals

ISO14649-1 contains the overview and fundamental principles for the standard. This part of the standard serves as an introduction and establishes the context and the domain of the standard. It does not contain any formal EXPRESS definitions and therefore does not include a machine readable component.

(ii) ISO14649-10: Generic process data (ISO2004)

Part 10 of the standard contains representation models for generic process data. The entities in this part of the standard are process and technology independent. Entities like manufacturing features, general program structure definitions and workinsteps are defined within the EXPRESS schema for this part of the standard.

(iii) ISO14649-parts 11 (ISO2004a), 12 (ISO2005), 13 & 14 : Process specific data

ISO14649-parts 11, 12, 13 and 14 contain constructs for representation of process specific data related to milling, turning, wire-EDM and sink-EDM respectively. These parts of the

standard contain entities describing the manufacturing process. They can thus be utilised to represent generic process plans for a product given that the process used to manufacture the part is known. Parts 13 and 14 are still in draft stages.

(iv) ISO14649-parts 111 and 121 (ISO2004b, ISO2005a): Tools

ISO14649-parts 111 and 121 contain entities to describe manufacturing tools for milling and turning respectively. Currently part 121 has been released as an international standard and part 111 is a final draft.

(v) ISO14649 part 16: Inspection

ISO14649-part 16 embodies the process specific data for measurement operations and is currently under development.

3.3.4. Knowledge Interchange Format (KIF)

KIF (Genesereth and Fikes, 1992) is a computer oriented language similar to first order logic that was created as an "Interlingua" or a mediator in translation of other computer languages. In most implementations it is supported by KQML. KIF has declarative semantics; the meaning of expressions can be understood without appeal to an interpreter. The language is logically comprehensive and at its most general allows expression of arbitrary logical sentences.

3.3.5. Extensible Mark-up Language (XML)

XML has become the de facto standard for structured data on the web (Goldfarb and Prescod 2004). It is an extension of SGML and was originally developed to meet the requirements for electronic publishing. XML provides the syntactic foundation to create XML-based markup languages. These foundations, similar in function to EXPRESS for STEP, include Document Type Definition (DTD), XML Schema and schema languages standardised under ISO/IEC 19757 commonly known as Document Schema Definition Languages (DSDL).

3.3.6. Common Object Request Broker Architecture (CORBA)

CORBA (Vinoski 1997) a standard defined by the Object Management Group (OMG) enables software applications to interoperate regardless of the language that was used to develop them.

An Interface Definition Language (IDL) is used to specify the interfaces for the various objects and their mappings to their native programming language.

3.3.7. Web Services

Web services (Newcomer 2002) are XML compliant applications mapped to software applications, objects or databases. Originally Simple Object Access Protocol and later Service Oriented Architecture Protocol and now simply SOAP is the Web services protocol for information exchange. The Web Services Description Language (WSDL) is utilised to describe the services and the XML based Universal Description Discovery and Integration (UDDI) for listing and discovering services. Web services are language and resource independent and provide a valuable framework for creating interoperable machine to machine interaction.

3.3.8. Knowledge Query and Manipulation Language (KQML)

KQML is a message handling protocol and a message format for knowledge sharing in agent based systems. It is based on the speech-act theory and describes "performatives" or operations that agents perform on each other's knowledge as well as the knowledge content. KQML can be used in conjunction with KIF to enable knowledge sharing among computer systems.

3.4. CAx Interoperability: The State-of-the-Art

The current body of research on interoperability is a subset of the CAD/CAM/CNC integration work. In this section an overview of the literature together with a categorisation and critique of the methods utilised by the researchers is provided.

3.4.1. STEP-AP238 and ISO14649 Based Solutions

STEP-NC provides an information exchange framework that when implemented by CAx systems can enable interoperability between the various resources.

(i) Investigation on STEP-NC as an integrator

Considerable research has been conducted to ascertain the effects of adoption of STEP-NC as the new manufacturing standard to replace G&M Codes.

Weck et. al (2001) provided an overview of the ISO14649 standard with regards to milling and the changes it could bring to the manner in which CNC machines are programmed. The main advantages of STEP-NC implementations identified in this early paper were bidirectional information transfer, easier object-oriented programming of CNC machines and feature-based NC programming.

Hardwick (2004) reviewed the data exchange standards for CAx systems including CAD, CAE, CAM, CNC and PDM and positioned STEP-NC within this context: to allow exchange of manufacturing information in a similar manner to which CAD information had been transferred with IGES and STEP. As a practical application, the research was applied to shipyard manufacturing processes to investigate the possible improvements (Hardwick et al. 2005). It was determined that by utilising STEP-NC the time for programming, setup and machining can be reduced significantly.

Wang and Xu (2004) investigated the application of ISO14649 and AP238 as an "adapter" for linking CAPP with CNC. AP238 was used as the information model for "generic STEP-NC information" that is hardware-independent and thus provides interoperability. ISO14649 was then employed to convey "machine specific" manufacturing information. The research was later expanded by Wang et al. (2006) with implementation considerations.

Zhang et al (2006) proposed a futuristic vision for an autonomous STEP-NC controller. Their design was based on adaptive and cooperative control architecture for distributed manufacturing systems (ADACOR) agents (Leitao et al. 2005 and Leitao and Restivo 2006). Within the proposed framework the controller could autonomously make manufacturing decisions based on its configuration. A complete implementation of such a controller could enable interoperability within the domain of CNC machines, as controllers can make individual decisions based on similar input.

Gang et al. (2006) presented XML as the suitable file format for carrying STEP-NC information on the web. Globally distributed manufacturing is one of the essential benefits of having interoperability as products can be designed and manufactured anywhere in the world, as such having a reliable and robust data transfer format is necessary for full realisation of

interoperability. The authors further determined that XML would be the suitable medium for the information models provided by STEP-NC to create an e-manufacturing platform. Similarly Borselino et al. (2004) used an AP238 implementation to demonstrate remote operation of an XML based procedure for Internet based CNC machining. Du et al. (2005) also proposed a comparable structure for an integrated CAD/CAM/CNC system.

Kramer et al. (2006) compared the performance of ISO14649 against AP238 when interpreted online and came to the conclusion that either standard can be used for development and they are interchangeable. It was their conclusion that with the current software interpreting AP238 takes a significantly longer time.

Xu (2006a) investigated the role of STEP-NC in the product development chain and how it can be used to convey generic and interoperable manufacturing information downstream from the CAD system to the CNC controllers. A web-enabled STEP-NC compliant manufacturing framework was demonstrated as a case study.

Table 3.4 presents a categorisation of these papers based on the technologies and standards employed by the researchers.

(ii) STEP-NC research with focus on milling technology

The major body of STEP-NC research can be categorised based on the research's main focus in terms of manufacturing technology and processes. The most popular area of research has been milling. Suh et al. (1995) started their integration research by retrofitting a CNC machine with a PC based controller to allow graphical simulation, a rarity at the time, and direct machining with no G&M codes. Hardwick (2002) provided one of the first visions of STEP-NC compliant manufacturing with regards to milling technology. The research was supported by Hardwick's earlier work on virtual enterprises (Hardwick et al. 2000).

Based on Hardwick's research Venkatesh et al. (2005) utilised tool centre programming (TCP) provided by AP238 in a joint effort between Boeing and NIST to illustrate an interoperable manufacturing scenario. Hardwick and Loffredo (2006) published a paper on this implementation documenting the presentation with 4 CAM vendors and 2 CNC controls on

two 5-axis machines with different axes configurations. Cutter centre locations were utilised to interpret a single AP238 file for the two machines. While this demonstrates interoperability, the fact that the AP238 file needs to be processed and converted into a controller specific format for each machine, reduces the flexibility of this approach to that of post-processors.

Table 3.4 - Research on STEP-NC as an integrator

Paper Reference	Technology			Standard		Data Storage	
	Mill	Turn	Other	AP238	ISO14649	XML	TEXT
Weck et al. 2001	☑				☑		☑
Hardwick 2004				☑			
Hardwick et al. 2005				☑			
Wang and Xu 2004				☑	☑		
Wang et al. 2006				☑	☑		
Zhang et al. 2006					☑		
Gang et al 2006					☑	☑	
Borselino et al. 2004				☑			
Du et al. 2005				☑		☑	
Kramer et al 2006				☑	☑		☑
Xu 2006a					☑		

Suh and Cheon (2002) suggested a framework for intelligent CNC based on ISO 14649 illustrated in Figure 3.2. This included the framework for a Shop-Floor programming system. This framework was extended to include an implementation method for a milling machine controller (Suh et al. 2002a). The modular design relied on CORBA as the facilitator in internal information exchange. The tool path generator was presented as a flow chart that checked for resources while interpreting the STEP-NC code automatically. This implementation was then realised in the form a STEP-NC compliant prototype milling machine (Suh et. al. 2002b). Suh et al. (2003) then designed and implemented the complete shop-floor programming system based on STEP-NC. The system can recognise the features in

an AP203 file and then generate ISO14649 manufacturing features. A process plan is then generated and finally the complete STEP-NC file is presented as the output.

Newman et al. (2003) provided a view on how the development of ISO14649 and ISO10303-238 can affect the evolution of CAD/CAM systems. An agent based CAM system capable of generating STEP-NC milling code was used to demonstrate the capability of STEP-NC in manufacturing prismatic parts.

Lee and Bang (2003) designed and implemented an ISO14649-compliant CNC milling machine. A proprietary XML coding of the STEP-NC file (i.e. non-STEP-XML) was used as the input for the CNC controller, a PC attached to a Motion Controller and through there to the stepper motors on the machine. The controller then interpreted the file and toolpaths were generated internally for the various workingsteps. Lee et al. (2006) later expanded the paper to present the work on an embedded controller illustrated in Figure 3.3.

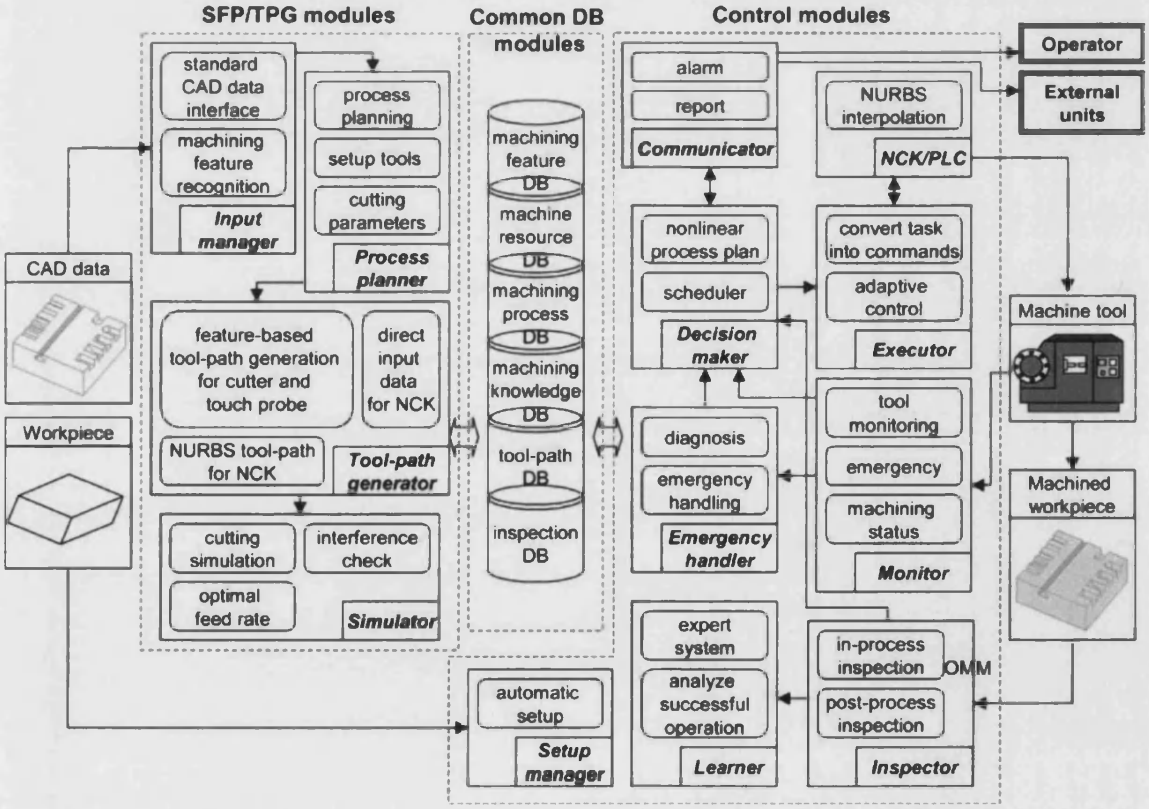


Figure 3.2 - A framework for intelligent CNC based on STEP-NC (Suh and Cheon 2002)

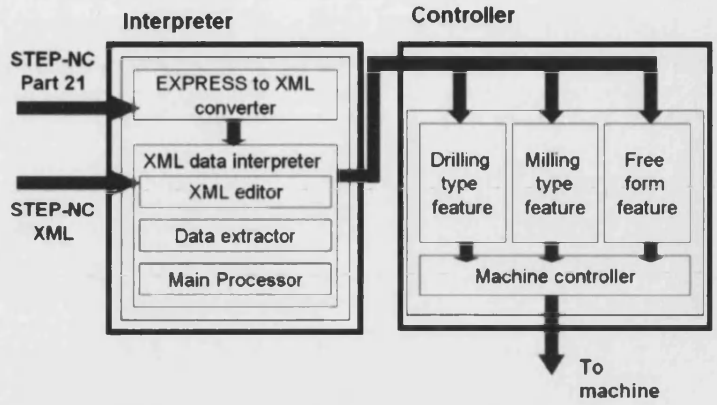


Figure 3.3 - Overall diagram for an ISO14649 compliant CNC milling machine (Lee et al. 2006)

Allen et al. (2005) utilised agent technology to develop an automated ISO14649 compliant process planning system for prismatic components entitled AB-CAM. In the presented approach a hierarchical decision making system was in place. Whenever a decision could not be made automatically, AB-CAM would ask the user to provide input. Such input however was not stored to provide a basis for future decision making.

Liu et al. (2006) presented a framework for an AP238 STEP-NC controller for milling machines. The controller was comprised of four modules: an interpretation module, a planning module, a simulation module and a CNC kernel. In the paper, only the interpretation module was implemented using the SDAI C++ bindings. The AP238 interpreter reads the STEP compliant data into native data structures. It is noteworthy that only the ARM constructs (i.e. ISO14649) where actually populated in the native data structures.

Xu et al. (2006) compared STEP-NC and function blocks (IEC 1999) as two methods for interoperable milling. Advantages and disadvantages of the two approaches with regards to the different aspects of interoperability were then documented. It was identified that no research has combined function blocks with STEP-NC.

Fichtner et al. (2006) created an agent based system for interpretation and machining of STEP-NC part programmes. The system based on cooperative agents can monitor information on the shopfloor while machining is taking place, and represent information from different local knowledgebases.

Amaitik and Engin Kilic (2007) developed an intelligent process planning system based on ISO14649. The system employs an inference engine based on a hybrid artificial intelligence approach composed of neural networks, fuzzy logic and rule-based decision making systems. The input is in ISO10303-224 format and an ISO14649 compliant XML process plan is generated.

The milling process has been one of the most popular technologies among STEP-NC researchers. Table 3.5 provides a summary of the research conducted with milling as its main focus.

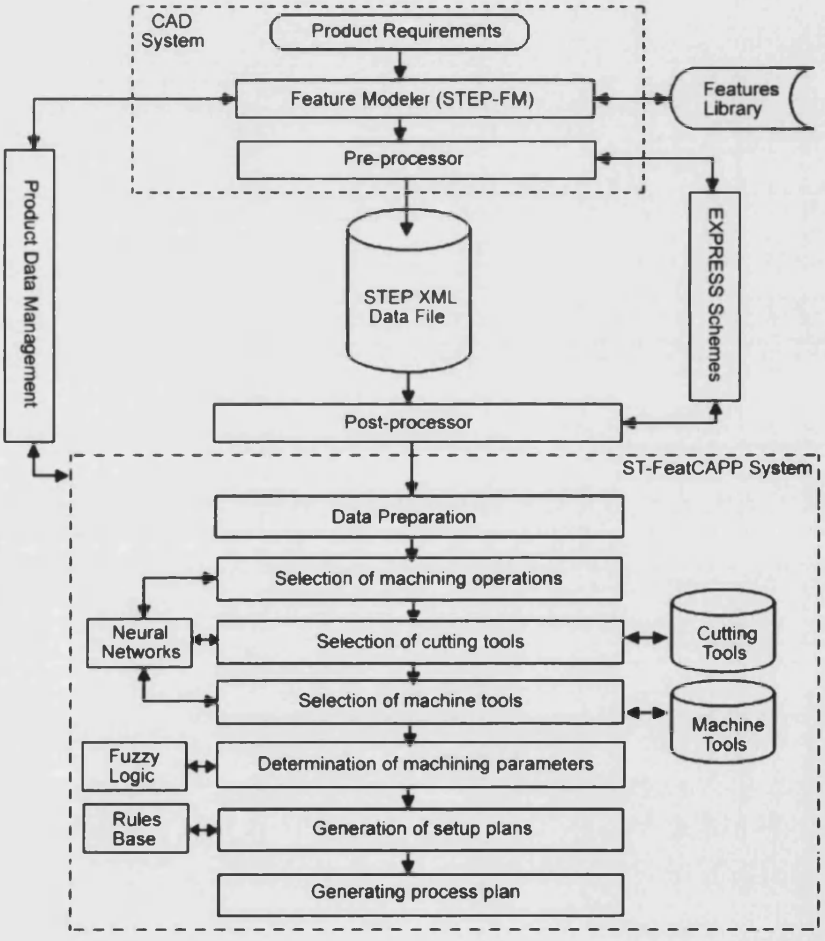


Figure 3.4 - Intelligent process planning system based on STEP-NC (Amaitik and Engin Kilic 2007)

Table 3.5 - STEP-NC research with focus on milling technology

Paper Reference	Technology			Standard		Data Storage	
	Mill	Turn	Other	AP238	ISO14649	XML	TEXT
Suh et al. 1995	☑						
Hardwick 2002	☑			☑			
Venkatesh et al. 2005	☑			☑			
Hardwick and Loffredo 2006	☑			☑			☑
Suh and Cheon 2002	☑				☑		
Suh et al. 2002a	☑				☑		
Suh et al. 2002b	☑				☑		
Suh et al. 2003	☑				☑		☑
Newman et. al. 2003	☑				☑		☑
Lee and Bang 2003	☑				☑	☑	
Lee et al. 2006	☑				☑	☑	
Allen et al 2005	☑				☑		☑
Liu et. al 2006	☑			☑			☑
Xu et al. 2006	☑				☑		
Fichtner et al. 2006	☑				☑		
Amaitik and Engin Kilic 2007	☑				☑	☑	

(iii) STEP-NC research with focus on turning and turn/mill

Turning and turn-mill machining STEP-NC oriented research has also been conducted around the world: Rosso et al. (2004) investigated the use of STEP-NC in manufacturing of asymmetric rotational components. It was the conclusion that ISO14649 part 10 features are capable of supporting the features these complex components require.

Xu and Wang (2004) developed a G-Code free lathe based on STEP-NC. Xu (2006) describes the approach where the STEP-NC file is converted into 6K programs (a machine native

format) and passed on to the retrofitted lathe. This low-level language, while not G&M codes, is still low-level and interpretation of STEP-NC code into this axis movement language is not that different to translating STEP-NC into G&M Codes.

Chen et al. (2005) proposed an RTCORBA based soft-bus to realise a framework for turning based on STEP-NC. The CORBA oriented data transfer would enable generic definition of STEP-NC resources.

Suh et al. (2006) proposed a novel architecture for an intelligent turning CNC controller that can interpret STEP-NC. The architecture is similar to those developed for milling by the same team. Choi et al. (2006) presented the implementation in the form of TurnSTEP, a STEP-Compliant CNC system for turning. In the paper, an XML schema was defined and mapped to the EXPRESS schema to support web-based manufacturing. The XML schema does not appear to be STEP-XML (ISO10303-28) compliant.

Heusinger et al. (2006) present a methodology for implementation of a CAx chain for rotational asymmetric parts. The necessary data models were created and tested through a prototype system.

Shin et al (2007) investigated the translation of G-Code programs written for lathes into STEP-NC compliant code. Utilising extra information such as the original CAD model, tooling information and machine instruction schema, algorithms for deriving geometric features, operations, etc were provided. A complete implementation of this research together with the earlier STEP-NC to G&M Code interpretations can be combined to allow non STEP-NC compliant machines to be interoperable with each other.

Table 3.6 enumerates the STEP-NC research with the main focus on turning and turn/mill

(iv) Other technology specific STEP-NC research

Inspection and online monitoring of the machining process has been another area of interest in the STEP-NC research. One of the first frameworks for STEP-NC based inspection was in geometric error measurement of spiral bevel gears (Suh et al. 2002c). While the paper does not

present any STEP-NC specific material, it does suggest that the on-line inspection can be added to ISO14649.

Ali et al. (2005) developed a STEP-compliant inspection framework for discrete components where an interoperable inspection process plan would be interpreted for on-machine probing as well as probing on a CMM.

Table 3.6 - Turning and Turn/Mill STEP-NC research

Paper Reference	Technology			Standard		Data Storage	
	Mill	Turn	Other	AP238	ISO14649	XML	TEXT
Rosso et al. 2004		☑			☑		☑
Xu and Wang 2004		☑			☑		☑
Xu 2006		☑			☑		☑
Chen et al. 2005		☑		☑			
Suh et al. 2006		☑			☑	☑	☑
Choi et al. 2006		☑			☑	☑	
Heusinger et al. 2006		☑			☑		☑
Shin et al 2007		☑			☑		☑

Brecher et al. (2006) positioned STEP-NC ISO14649-16, the inspection part of STEP-NC within the domain of inspection standards such as ISO10303-219, Dimensional Measuring Interface Standards (DMIS), dimensional markup language (DML) among others. Part 16 of STEP-NC was then utilised to create a closed-loop inspection system. The results were however stored in additional entities in the original STEP-NC file. When a single STEP-NC file is used to manufacture a large batch of parts and measurements have to be put back into the file that already contains manufacturing information creates a lot of redundancy.

Wosnik et al. (2006) suggested an approach for enabling feedback of process data in a STEP-NC compliant machining facility. The ISO14649 was extended to accommodate the extra entities that were needed to support the information model for online monitoring of the

execution of the workingsteps (i.e. whether a drilling workingstep was completed successfully).

Zhao et al. (2007) extended the on-line STEP-NC compliant research by supporting real-time, closed-loop machining with the integration of machining and inspection workingsteps within the same STEP-NC program.

There have been a number of STEP-NC papers on other processes as well. Garrido Campos and Hardwick (2006) defined a traceability model for CNC manufacturing based on an extension to AP238. It is proposed that traceability data be stored in separate files for each instance of the manufactured product.

The files would be ISO10303-21 compliant files based on a new EXPRESS schema designed by the authors. The traceability file is linked to the manufacturing process file through the use of hash numbers in the manufacturing file. This approach represents a problem with maintaining the integrity of the information links. If one of the files is modified, the link between the files is severed and both files need to be monitored simultaneously to ensure that the information is valid.

Sokolov et al. (2006) described the implementation of wire electro discharge machining (EDM) algorithms for STEP-NC. Ho et al. (2005) presented an information model for wire electrical discharge machining in compliance with STEP-NC ISO14649-13. The research was tested by using a prototype system supported by Java and an Object Oriented Database Managements System (OODBMS).

Ryou et al (2006) proposed an EXPRESS data model as an extension to ISO14649 to represent layered manufacturing (LM) processes. The proposed model however does not utilise the existing manufacturing features in ISO14649 and ignores program control entities such as workplan and workingsteps.

Bi et al. (2006) introduced a new type of CNC named Intelligent Integrated Numerical Control (I^2NC) that proposed an approach to create a unified platform oriented to CNC manufacturing. The implementation of the platform would allow the move from distributed work to

collaborative manufacturing. The conceptual architecture of the platform is illustrated in Figure 3.5. A prototype of the platform was implemented on a roll grinder machine.

Stroud and Xirouchakis (2006) investigated the use of STEP and STEP-NC in the manufacture of aesthetic shapes. The manufacturing of aesthetic products in kitchens and bathrooms, architecture, gravestones, gardens and lamps was discussed and the relevance of the STEP-NC model to their manufacturing was surmised. As the features in these types of applications are considerably different to those of industrial applications, STEP-AP224 was deemed as insufficient for representation of the geometry and additional EXPRESS entities were proposed in the research.

Table 3.7 provides an overview of other technology specific STEP-NC research.

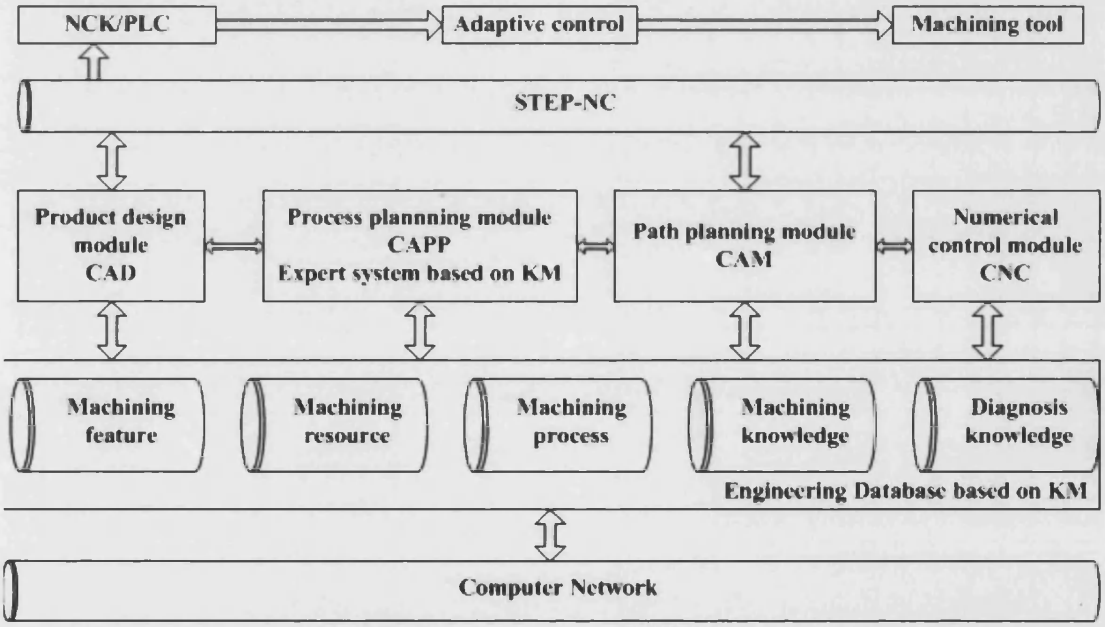


Figure 3.5 - Conceptual architecture of I^2NC

Table 3.7 - STEP-NC research with focus on technology other than milling and turning

Paper Reference	Technology			Standard		Data Storage	
	Mill	Turn	Other	AP238	ISO14649	XML	TEXT
Suh et al. 2002c			Inspection		☑		
Ali et al. 2005			Inspection		☑		☑
Brecher et al. 2006			Inspection		☑		
Wosnik et al. 2006			Inspection		☑		
Zhao et al. 2007			Inspection		☑		☑
Garrido Campos and Hardwick 2006			Traceability	☑			☑
Sokolov et al. 2006			WEDM		☑		
Ho et al. 2005			WEDM		☑		☑
Ryou et al 2006			Layered Manuf.		☑		
Bi and You 2006			Roll Grinding		☑		☑
Stroud and Xirouchakis 2006			Aesthetic shapes		☑		

(v) STEP-NC review papers

A number of review papers documenting the advances in STEP-NC across different manufacturing processes and within different research teams have been published since the development of the standards. Xu and He (2002) ascertained the effect STEP-NC would have on manufacturing in the coming years. Later Xu and He (2004) reviewed the ongoing research on STEP-NC and categorised the challenges the standard would face. Xu et al. (2005) followed the earlier review paper with a more comprehensive review of the STEP-NC research effort that documented the two IMS projects undertaken on the subject. Xu and Newman (2006) carried out a review of the technologies to make CNC machine tools more open, interoperable and intelligent. In the paper, the above mentioned presentation was discussed with issues with the lack of implementation of machining features and design data recognised.

Saaski et al. (2005) provided a review of the STEP-NC research and investigated the results within the context of realistic manufacturing scenarios. Table 3.8 presents these papers and categorises them based on the manufacturing technology in the paper, the specific part of the standard used (ISO14649 ARM or ISO10303-238 AIM) and data storage standard utilised (XML or Text).

Table 3.8 - STEP-NC reviews

Paper Reference	Technology			Standard		Data Storage	
	Mill	Turn	Other	AP238	ISO14649	XML	TEXT
Xu and He 2002	☑	☑			☑		☑
Xu and He 2004	☑	☑	Other Techs Reviewed	☑	☑		
Xu et al. 2005	☑	☑	Other Techs Reviewed	☑	☑		
Xu and Newman 2006	☑	☑		☑	☑		☑
Saaski et al. 2005				☑			

(vi) STEP-NC in industrial publications

STEP-NC generated some early enthusiasm in the industrial publications: Mathews (2002) cited 75% performance increase and 50% time benefits for small job-shops. Hardwick (2002a) mentioned similar numbers. Stevens (2003) reviewed STEP-Tools' products for STEP-NC Development marking them as the first tools available for early adopters. Newman (2004) described the shift from G and M codes' "How to Manufacture" paradigm to "What to Manufacture" of STEP-NC. Hoske (2006) reported on the migration from axis movement programming to cutter movement programming based on AP238 and its effects on interoperability.

As STEP-NC is a standard for information representation it is only as useful as the resources that adhere to its guidelines and employ its protocols for data exchange. The lack of industrial support for the standard however, has limited the actual implementations to research environments. The commercial scepticism has probably been reinforced due to STEP-NC

being a feature-based standard as industrialists often find features limiting in representing complex products with sculptured surfaces. Callen (2005) considers STEP-NC as "an advancement still searching for its markets".

3.4.2. Agent Based Manufacturing Interoperability

Distributed artificial intelligence techniques can be used to transfer and manipulate information throughout the CAx chain and construct the distributed knowledge-base required for the intelligent interoperable integration of product data models and manufacturing resources. Intelligent agents are the practical implementation of distributed artificial intelligence methodology. Each individual intelligent agent is an intelligent entity that is autonomous to a certain degree, has sensors to sense its surrounding environment, has a goal or agenda to pursue, and has effectors to make changes in the environment (Woolridge 2002). Agents have been categorised according to their functionality by researchers. Nwana (1996) provided an extensive typology of intelligent agents that can be seen in Table 3.9 with the description provided for each type of agent.

Table 3.9 - Software agent typology proposed by Nwana (1996)

Software Agent Type	Description
Collaborative agents	Autonomy and cooperation are the main characteristics. Learning might be present but is not the main emphasis.
Interface agents	Learning and autonomy are the main focuses.
Mobile agents	Agents capable of migrating to different parts of a computer network to achieve their function.
Information agents	Manage and gather information from many data sources.
Reactive Agents	Agents that only respond to a specific stimulus and are not generally aware of their environment.
Hybrid agents	Agents that have a combination of above mentioned abilities.
Heterogeneous Agents	Integrated set of different kinds of agents that can function together. Some agents might be hybrid agents.
Smart Agents	Agents that have a grasp of reality and can make decisions while adapting to the changing environment. Impossible to distinguish with a human element. Nwana believes that these agents are just aspirations of researchers and do not exist in reality.

While individual agents are extremely effective at performing tasks like gathering information, the real potential of agent systems is realised when they are combined in Multi-Agent Systems. Usually different types of agents are combined in the Multi-Agent Systems to achieve better functionality. These systems are very effective in solving complex problems where the solution can emerge from solving a great number of smaller problems linked together. Individual agents try to solve the small components of the problem while maintaining the feasibility and integrity of the solution through communication with their peers.

Taking the multi-agent concept one step further are the mobile agent systems. In these systems the intelligent agents reside on different computers connected through a network or the Internet and are able to move from one system to the other when instructed to do so. The mobile agents can preserve their state through the utilisation of persistent contexts. In other words, the mobile agents retain the values of all their variables when transferred and can therefore continue execution when moved from one computer system to another (Picco 2001).

Agent technology has been utilised in interoperable CAx manufacturing research by a number of researchers:

Wang et al. (1998) surmised that the next-generation of intelligent manufacturing systems will be multi-agent systems with distributed control mechanisms. Function blocks are used to design the next generation of manufacturing control systems. The designed multi-agent based system is device independent and as such could be used to control machines in an interoperable manner.

Peng et al. (1999) proposed a multi-agent system for intelligent manufacturing enterprise integration. Within the system a set of agents with specialised expertise could be assembled rapidly to gather the relevant information and knowledge. The research was done within the framework of *Consortium for Intelligent Integrated Manufacturing Planning Execution* (CIIMPLEX). The vision for this consortium is to create an open, distributed, interoperable integrated system for manufacturing planning and execution.

Zhao et al. (2001) identified the interoperability issue in the design domain and devised a multi-agent based cooperative design environment called CLOVER to enable high level dynamic and autonomous cooperation among applications. A prototype implementation of the environment was utilised to demonstrate the interoperability advantages.

Zhou et al. (2002) developed a model for a distributed internet-based scheduling system for the manufacturing chains based on Aglets. In their model, scheduling is handled in two levels. In high-level scheduling, Aglets support the operation of manufacturing venues and cells. In low-level scheduling, working procedures over manufacturing equipments are decided by Aglets. Based on this research Zhou and Jiang (2005) later used the mobile agents to encapsulate manufacturing resources to allow them to communicate over the internet. This approach allowed legacy CAD/CAM and CNC software to communicate via an Internet Protocol (IP) network. A wide implementation would allow raw data exchange between different CAx systems and serve as the information exchange foundation for interoperability.

Wang (2002) and later Wang and Shen (2003) proposed a distributed process planning (DPP) system based on agents and function blocks. The event flow between the function blocks determines the process plan and based on the CNC machine available, the machining strategy to be applied and other information is interpreted by the CNC controller. It should be noted that it is required for the controller to be open to allow function blocks to be processed locally.

Shin and Jung (2004) developed a mechanism called mobile agent-based negotiation process (MANPro) to support the creation of a distributed shop floor control system. In this mechanism the control is achieved through negotiations between autonomous agents. The agents are aware of the resources they represent and make bids based on this information.

Shen et al. (2005) proposed the framework for *iShopFloor,* an intelligent agent based architecture to connect CNC manufacturing resources to each other on a computer network. The XML based information exchange architecture used through the research for data transfer is extensible and has potential for supporting interoperability.

A comprehensive review of applications of agent-based systems in intelligent manufacturing was conducted by Shen and Norrie (1999) and updated to cover the recent advances by Shen et al. (2006).

3.4.3. Computer Aided Process Planning in Relation to Interoperability

CAPP is a major topic of research in the CAx manufacturing domain. A great number of publications have focused on different aspects of CAPP. Some researchers however, have identified CAPP as the main enabler for interoperability in the CAD/CAM/CNC chain and studied interoperability as the main advantage of implementation of CAPP.

ElMaragy (1993) identified computer aided process planning (CAPP) as the prominent enabler of achieving seamless integration between the various CAx systems. The need to bridge the gap between CAPP and production planning and control (PPC) was highlighted in the paper and three strategies for eliminating this gap were provided: global integration, unification and modular integration. In global integration, each CAx system would maintain its own database and an update scheme is utilised to ensure the integrity of information throughout the enterprise. In unification, CAPP and PPC would be combined into a single system within a single structure. In modular integration, an intermediate between the first two strategies, a separate "integrator module" is created to bridge the functional and data gap between CAPP and PPC.

Srinivasan et al. (1996) explored direct interfacing of CAD systems with process planning software through the use of a commercial CAD package and a commercial process planning system. In the research the database system for an integrated process planning tool was specified and the required information repositories were identified as machine database, material database, fixture database, tool database and cutting parameter database.

Miao et al. (2002) demonstrated the use of features in automating process planning tasks and integration of CAD and CAM. A system was proposed for recognising features in a CAD generated STEP file to convert faces into manufacturing features. These features are utilised in generation of an automatic process plan. The process plan is subsequently transferred to a CAM system where toolpaths are generated.

Dereli and Baykasoglu (2005) demonstrated an open and optimised process planning system for prismatic parts names "OPPS-PRI 2.0" to increase the responsiveness of SMEs to market changes. In the research, process planning was viewed as the main enabler of CAD/CAM integration. The proposed process planner recognises features contained in a STEP formatted file and after a series of automated planning operations generates and verifies G&M Codes for CNC machining.

Ni et al. (2006) presented an integrated process planning system to assist SMEs in achieving concurrent engineering in design, process planning and manufacturing. A level-based manufacturing resource model and a feature-based part information model were developed to facilitate integration and decision-making. A capability coding schema was then utilised in process planning together with a semantic rule configuration technique to generate NC code.

Wang et al (2006a) utilised function blocks as reusable coding devices to encapsulate process plans. A two layer system was used to support generic and machine specific process planning. As the generic process plan was utilised to create machine specific process plans, Function blocks evolved from Meta function blocks containing generic information to execution function blocks with machine specific information such as depth of cut and feed rate.

3.4.4. Manufacturing Interoperability Based on Enterprise Integration

Twigg et al. (1992) studied the implications of CAD/CAM systems on integration. In this research three integration approaches in manufacturing enterprises were analysed: integration by merging functions of different sections, integration through a distributed or centralised database for production information and integration through the establishment of linkage mechanisms between functions. These linkage mechanisms are essentially enablers of interoperability for information systems supporting different functions in the manufacturing company.

Mejabi and Singh (1997) devised a framework for enterprise-wide integration in manufacturing companies. The relationship between various architectures for integration namely, data architecture, network architecture, decision making and control architecture and functional architecture served as the basis for the framework.

Alsene (1999) also conducted research in computer integration of the enterprise. In the research it was made clear that the author believed "enterprise integration" to be just a new term for "computer integration of the enterprise" and thus argued that mechanisms other than information technology such as social and business mechanisms are neglected by using the new terminology when referring to integration of manufacturing enterprises. A comprehensive overview of the terminology and definitions was provided followed by a case study.

Ortiz et al. (1999) created an integration framework based on The Purdue Enterprise Reference Architecture (PERA) and the Open System Architecture for Computer Integrated Manufacturing (CIMOSA) (Bernus 1996).

Mehrabi et al. (2000) outlined a definition of reconfigurable manufacturing systems and studied these systems' roles in future manufacturing and their effect on an enterprise's market responsiveness. A historical overview of the scientific understanding, engineering technology and shifts in the marketplace concluded with design criteria for reconfigurable systems of the future.

Bernard and Perry (2003) summarised the basic concepts of data integration in a product's lifecycle. The paper reviews the problems associated with modelling products and systems linked to process modelling and planning and identifies suitable modelling of information generated during a product's lifecycle as the main enabler for data integration and product lifecycle management. Gao et al. (2003) utilised product data management technologies to enable enterprise integration. The integration of conceptual designs and analysis with the enterprise resource planning (ERP) and manufacturing planning systems was done through the use of STEP AP224.

Chen and Vernadat (2004) presented a survey of the standards related to manufacturing enterprise integration standards. This is a follow up to the research conducted by Kosanke (1997) and discusses the various standards that relate to enterprise integration. Lower levels of integration such as those useful for enabling interoperability among individual CAx systems however, were not covered by the research and integration was considered on a broader system view.

Giachetti (2004) defined a framework to review enterprise integration. This framework was then utilised to analyse the used technologies to ascertain their strengths and weaknesses in various aspects of integration. In order to achieve this, the enterprise was studied in 4 layers as illustrated in figure 3.6 with the relevant type of integration for each layer identified. Interoperability was identified as the appropriate integration type for applications.

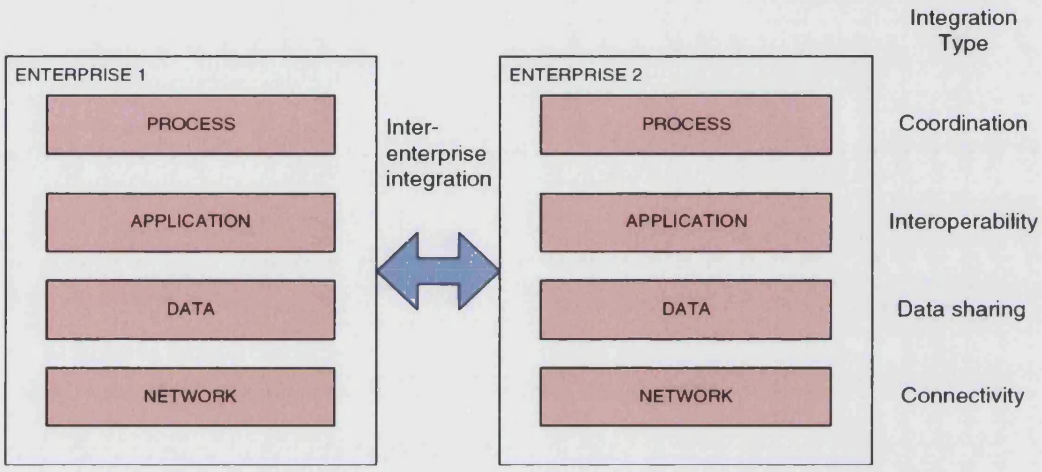


Figure 3.6 - Enterprise information integration framework (Giachetti 2004)

Smith (2004) identified portals as the main enablers for application interoperability. A portal was defined "as an infrastructure providing secure, customisable, personlisable, integrated access to dynamic content from a variety of sources, in a variety of source formats, wherever it is needed".

Dan et al. (2005) proposed a model of network-integrated manufacturing systems (NIMS). A systematic integration approach including information integration, process integration, knowledge integration and organisation integration was presented. The model was presented using 5 concentric rings representing - from centre to perimeter - core objectives, participants, support centres, functional subsystem and infrastructure.

Zaidat et al. (2005) presented a framework for organisation network engineering and integration. As seen in figure 3.7, it was suggested that the organisation network consists of a

number of enterprises and their interactions. The implementation of the framework was carried out for a network of SMEs that provide services to the petroleum industry.

Zhou et al. (2005) devised an adaptive model of the virtual enterprise based on web services. The model aimed to achieve sharing and integration of enterprise information resources. Web services registration and discovery methods were utilised to allow dynamic organisation of the virtual enterprise at run-time.

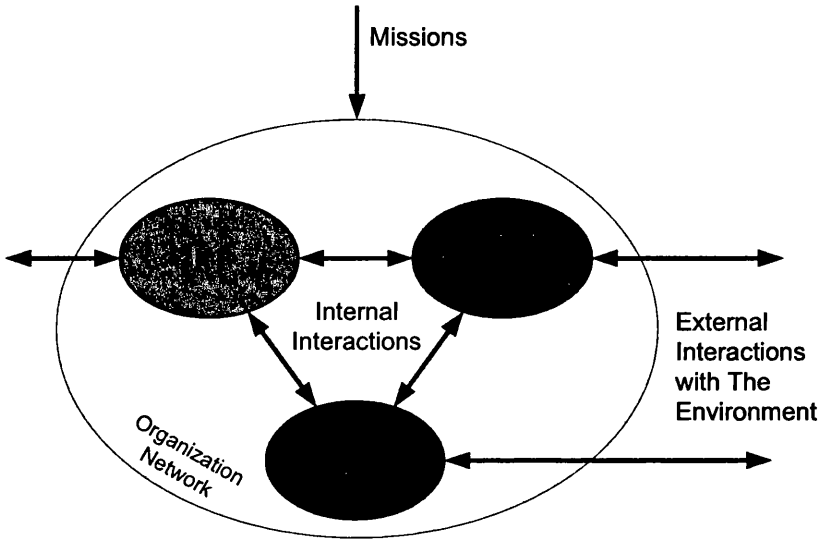


Figure 3.7 - Organisation Network (Adapted from Zaidat et al. 2005)

3.4.5. Semantic Interoperability

Semantic interoperability is ensuring that the same set of semantics is interpreted from the data that is transferred between two systems. While semantic interoperability has been a topic of interest for researchers in various fields including those working in enterprise integration and computer science, the interest from the manufacturing research community has been limited. Nevertheless semantic interoperability provides generic tools that can be applied to solve interoperability problems in the domain of CNC manufacturing.

Euzenat (2000) provided a preliminary report on semantic interoperability. The definition of languages, semantics and transformation was presented followed by identification of the

following levels of interoperability in an ascending order of capability: encoding, lexical, syntactic, semantic and semiotic.

Zimmerman et al. (2002) reported on feature-based product development. A Unified Model of Engineering Objects and Engineering Object Relations (UMEO) was then suggested to represent all application-specific, standardised and non-standardised Engineering Objects (EO) in a globally accessible taxonomy. Where an EO is any object relevant in engineering such as assemblies, features (i.e. machining features [MF], design features [DF] and quality assurance features [QAF]), parts, surfaces, tolerances, materials, etc. Figure 3.8 illustrates this concept.

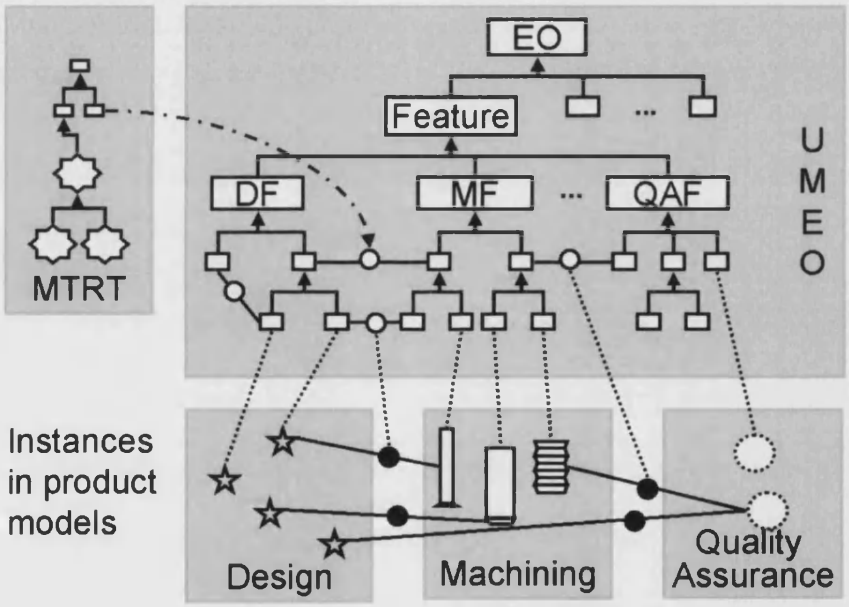


Figure 3.8 - UMEO classes and instances (Zimmermann et al. 2002)

Tolk and Muguira (2003) specified the various levels of conceptual interoperability. These levels are identified in Figure 3.9. At level 0 there is no interoperability and data is hard-coded into the source code of individual systems. At level 1 the data is documented and therefore the format is known. Level 2 adds the standardised models and makes sure that various systems are using a common ontology when representing information. Level 3 supports interoperability in the dynamic view of the systems as well as static and functional views

supported by the lower layers. Level 4 enables interoperability on the conceptual model level where the intent of information within each system as well as logical and conceptual relationship between the data contained within various systems are known.

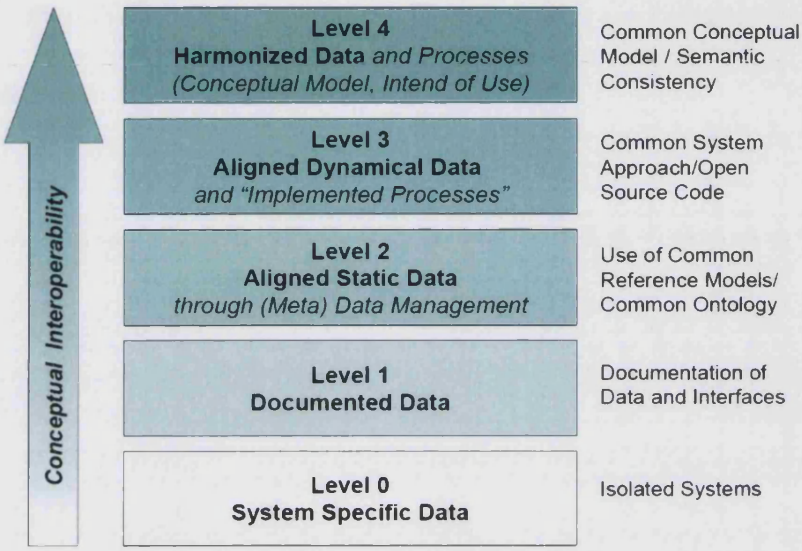


Figure 3.9 - Levels of conceptual interoperability (Tolk and Muguira 2003)

Lin et al. (2004) provided a Manufacturing Systems Engineering (MSE) ontology that enabled semantic interoperability among extended project teams. The ontology approach was chosen to define the MSE moderator and syntax and semantic issues in the interoperability domain were discussed.

Vetere and Lenzerini (2005) identified the management of semantic differences as the major challenge in providing interoperability in distributed, heterogeneous environments. To meet this challenge, four basic models for semantic interoperability were presented in the research together with the placement of the integration logic for each model.

Da Silva et al (2006) presented a three-step approach for enabling interoperability between heterogeneous semantic resources. The first step is to homogenise the semantic resource representation formats, the second is to align resources by creating a mapping of the semantic

resource entities and the final step is to rank the mappings obtained in the previous step to suggest a contextualised measure of the mappings.

3.4.6. Open and Intelligent CNC for Interoperability

A number of researchers have identified open and intelligent CNC controllers as the main enabler for interoperability. Pritschow et al. (2001) recognised open control systems as the key enabler for modular and re-configurable manufacturing systems. The paper provides an overview of the open control architecture and its effects on manufacturing. The criteria for estimation of the openness of a controller were portability, extendibility, interoperability and scalability. Figure 3.10 illustrates these criteria according to the authors.

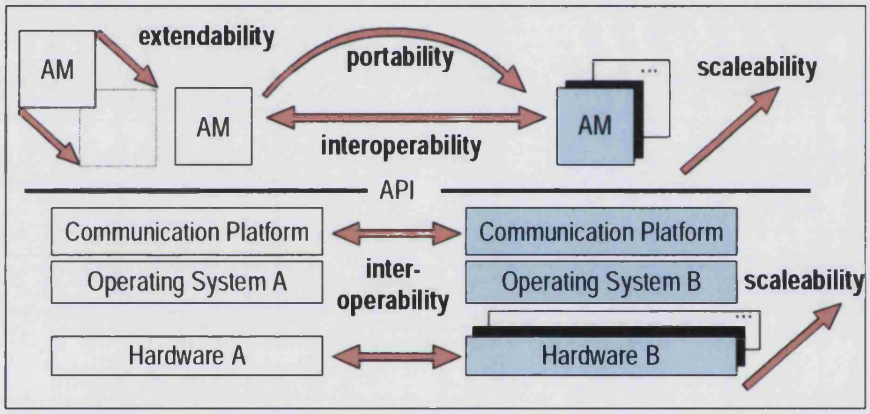


Figure 3.10 - Criteria for determining the openness of control systems (Pritschow et al. 2001)

Nacsa (2001) identified the research gaps for the application of artificial intelligence (AI) in intelligent CNC controllers. The areas where AI's effect would be more prominent were indicated and after a brief review of open controllers, the implications of utilising a knowledge server concept in the design of the CNC controller were discussed.

3.4.7. Remote Web-Based Control and Monitoring for Integration of CNCs

Researchers have recognised web-enabling of manufacturing resources as a means to better integrate them in a global enterprise leading to better interoperability. The ability to remotely monitor and integrate machining resources on the internet is the main focus of the research conducted by Wang et al. (2003). Java3D (Liang and Zhang, 2007) technology was used in conjunction with real-time sensor data to remotely monitor and control a machine tool on the

internet. A tripod machine was chosen to demonstrate the flexibility and extendibility of the approach undertaken in the paper. Wang et al. (2004) extended the research to create a framework to cover the remote control of a complete workshop. The framework called Wise-ShopFloor allows traditional CNC machines to be brought on-line with a combination of control and monitoring capabilities.

Ong et al. (2002) proposed a web-based virtual manufacturing (VM) system for milling based on Java and the Virtual Reality Modelling Language (VRML). The system provides an open environment that can be refined and modified for other uses. Three test case studies on collision detection, tool life and power consumption were used to demonstrate the advantages of the virtual system.

3.4.8. Manufacturing Interoperability by Low-Level Data Transformation

A few researchers proposed that low-level languages utilised in manufacturing can be translated into one another automatically thus enabling interoperability without requiring a higher level of information in the CAx chain.

Fortin et al (2004) suggested the use of an innovative language called 'Base Numerical Control Language (BNCL)'. The architecture of the proposed system was designed around the concept of a BNCL virtual machine (an interpreter) and BNCL virtual hardware (an abstraction of the machine). The approach is complementary to that of STEP-NC and is based on creating an open, interoperable abstraction of the machine to be programmed using a low-level assembly-like language.

Schroeder and Hoffman (2006) demonstrated a software tool that could automatically convert a G&M code program for one CNC to the format usable on another CNC. The system uses an external XML rule file to control the conversion process. It was reported that 90% of a complex NC program could be converted automatically with the remaining 10% requiring input from the user.

Liu et al. (2007) proposed and designed a system called NC program processor (NCPP) that parses a G&M Code RS274D compliant file and generates a simple representation of motion commands, programmable logic controller (PLC) commands or simple parameter settings

called canonical machining functions. The logic utilised to generate this representation could be expanded to determine certain semantics from NC files.

3.5. Critique

The research reviewed in the previous sections of this chapter outlines a wide range of techniques and approaches that have been adopted across the world to tackle the problem of the lack of interoperability in CNC manufacturing. There are a number of research gaps that impede the implementation of the suggested techniques in the industry. These include:

3.5.1. The Requirement for Modification of Current Resources

Every STEP-NC paper reviewed above (with the exception of Shin et al. 2007) assumes that the standard will be eventually adopted not only by machine vendors and controller manufacturers but also CAD/CAM developers. While this unification of standards would be greatly beneficial to end-users the author believes that the commercial gain imaginable for the suppliers of CAx resources is very limited.

Table 3.10 illustrates the changes that would happen in case of wide adoption of STEP-NC from the perspective of vendors and users. The table clearly shows that while the wide adoption of STEP-NC is very beneficial to the users in providing them with means to achieve greater flexibility and interoperability, CAx vendors do not benefit from this change.

If STEP-NC is adopted on a wide basis, CAD vendors will have to adapt their systems to support feature-based standards such as STEP-AP224. Otherwise feature recognition systems are required by the CAM developers to detect and identify manufacturing features in the product design. This process as mentioned in the sections above has been a topic of research for many years and no universally acceptable solution has been developed for it yet.

CAM vendors will also need to create significant changes in the way their software works. In addition to supporting features, they will need to adopt the process planning logic of STEP-NC (i.e. workplans and workingsteps) and allow the reading and writing of STEP compliant process plan files.

Table 3.10 - The effects of wide adoption of STEP-NC in manufacturing

	State-of-the-art	Wide adoption of STEP-NC
User	Has to contend with numerous standards to represent data for each resource	A unified standard will be utilised along the product lifecycle
	Buying a CNC machine with a different make to that of the existing machine pool will incur significant training costs	Every machine tool will interpret the same process plan. Extensive training for learning a new programming language will therefore not be required.
	The proprietary dialects of standards used by the resources hinders the ability to transfer information from one system to another	Information can be transferred from one resource to another with integrity. The unified standard would mean a single set of semantics across the chain and eliminate problems associated with interpretation differences.
CAD vendors	Use proprietary formats as well as non-feature-based standards such as STEP-AP203 and IGES.	CAD systems have to be modified to support STEP-AP224 feature-based geometry standards.
CAM vendors	Use proprietary standards to store manufacturing information and the process plan.	CAM systems have to be modified to support STEP-AP238 and ISO14649 to store manufacturing data.
	Have proprietary tool-path generation algorithms for specific types of products. A high value is usually associated with the intellectual property that constitutes these algorithms.	As tool-paths are not necessary in STEP-NC and the CNC controller is responsible for interpretation of the process plan, CAM vendors will lose the marketing value associated with their tool path generation algorithms.
	For each CAM-CNC combination a post-processor is required. Obtaining a new CAM system different to the existing software pool will require post-processors for all CNCs to be obtained as well.	Since the interface between all makes of CNC and CAM software is the same, CAM systems can be used interchangeably. Each CAM system will be assessed only based on the capabilities it provides and not based on its compatibility.
CNC manufacturers	Each CNC has its own proprietary dialect. If a company has a number of CNC machines, the most cost effective way for them to buy new machines is to buy the same brand to avoid training costs.	All CNC machines utilise STEP-NC. When buying new machines, any CNC can be considered regardless of make and model.

A great amount of development time by CAM vendors is spent on creating optimised tool-paths and process planning techniques for a specific range of products. As STEP-NC is resource independent and contains a generic process plan, this knowledge needs to be pushed

down to the controller. CAM vendors will therefore lose their marketing advantage in advertising superior tool-path generation algorithms.

For each CAM-CNC combination, a post processor is currently required. This postprocessor converts the process plan from product space to machine space and is therefore resource specific. This is illustrated in Figure 3.11. With the wide adoption of STEP-NC the requirement for these postprocessors will be eliminated as illustrated in Figure 3.12. CAM systems can therefore be utilised interchangeably and compatibility will no longer be an issue.

Considering the above facts, it is the author's opinion that unless users exert considerable pressure on the developers it is very unlikely that CAD/CAM/CNC vendors would start implementing STEP-NC on a wide basis.

With this in mind, the applicability of the STEP-NC research reviewed above to the industry is somewhat limited at the moment. An interoperable solution that can support heterogeneous CAx resources can eliminate this problem, as no changes will be required by the vendors. Therefore the same benefits as STEP-NC can be realised in a shorter time frame with such a solution.

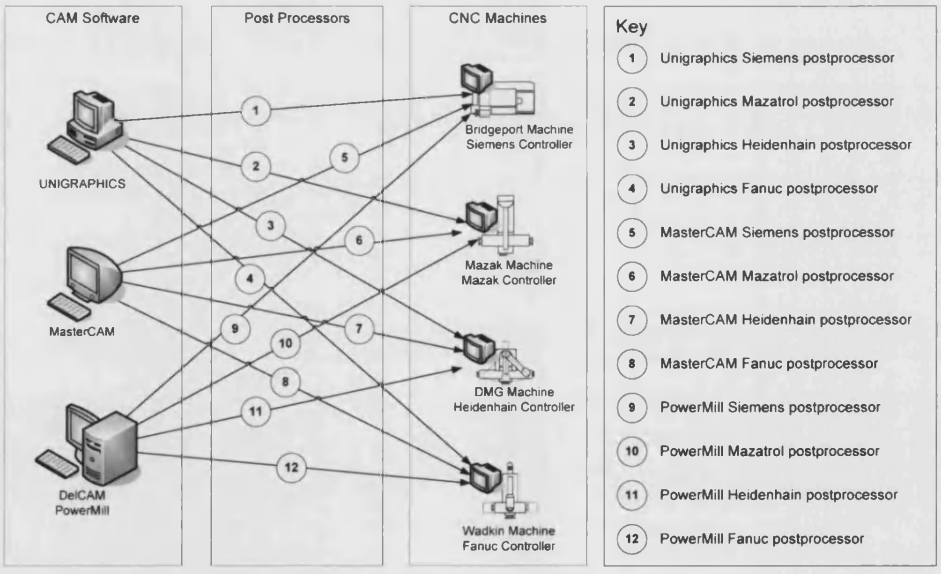


Figure 3.11 - CAM-CNC combinations for postprocessors

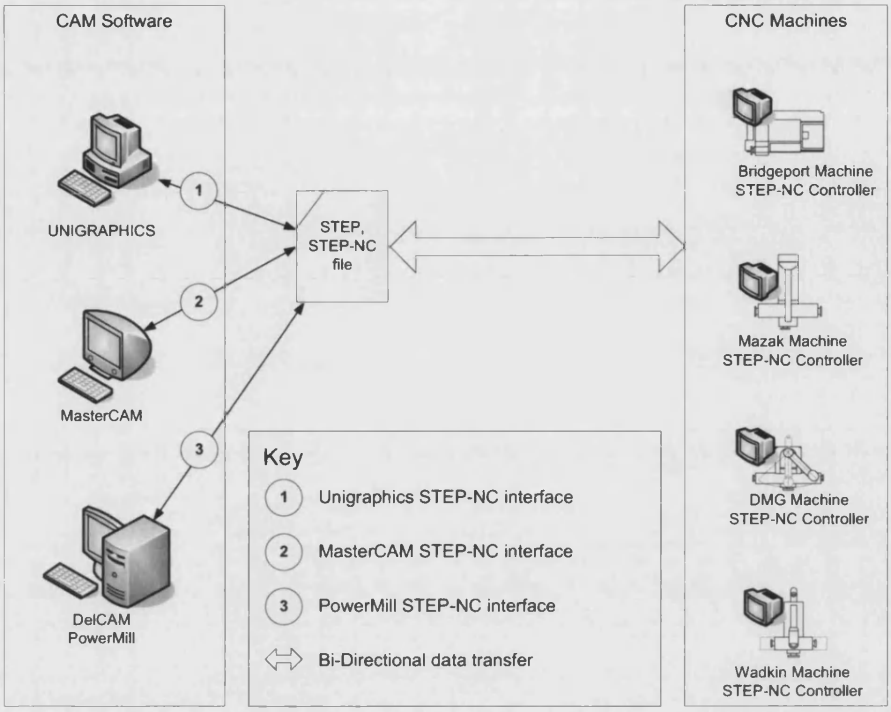


Figure 3.12 - The vision of STEP-NC replacing the postprocessors

3.5.2. Lack of Semantic Interoperability in CAx Manufacturing

Most of the semantic interoperability research reviewed in section 3.4.5 has been conducted in the computer science and enterprise integration domains. Currently the challenges in creating semantic interoperability between CAx resources are relatively unknown.

3.5.3. Reactive Research as Opposed to Proactive / Revolutionary Research

The research reviewed in the previous sections of this chapter presents a body of problem-solving efforts that have been conducted to eliminate specific problems within the domain of CAx manufacturing. It is the author's opinion that more revolutionary and proactive thinking is required in manufacturing research to maintain competitiveness of the manufacturing industry in the knowledge oriented market of today.

3.5.4. STEP-NC and Object-Orientation

Most of the literature above refers to STEP-NC as an "Object-Oriented" model for storing manufacturing information. According to Garrido (2003) every object needs to have a state, behaviour and identity. Objects are instances of classes. A class defines the abstract

characteristics of an object including its properties or attributes and its features or methods (i.e. the things that an object can do). While EXPRESS schemas in STEP define properties and attributes, they do not provide a mechanism for defining behaviours. This means that while some advantages of object-orientation such as inheritance are applicable to STEP-NC models others like polymorphism are impossible to implement due to lack of method definitions.

4. A Novel Object-Oriented Interoperable CAx Framework

4.1. Introduction

In this chapter a framework for realisation of interoperability in CNC based manufacturing supported by the Object Oriented methodology will be presented. The research gaps identified in the previous section will be used to develop a set of requirements for the framework. The requirements will then be utilised as blueprints to devise the necessary functionalities in the framework. The flow of information and interconnections between the various elements are then introduced and analysed.

4.2. Requirements for an Interoperable CAx Framework

The critique in section 3.5 together with the scope and the hypotheses of the research (see section 2.2) leads to the development of a set of requirements for a novel interoperable CAx framework. The interoperable framework should be able to allow the interchange of information among the CAx systems while maintaining the integrity of the information. The following requirements have been determined for the development of an effective framework:

4.2.1. Standardisation

A proprietary enabler for interoperability is not always useful in the long-term as any development needs to be carried out by the original inventors. A standardised system on the other hand could benefit from a multitude of developers and supporters. Selection of open standards could broaden this even further by allowing the open source community to contribute to the framework in the future. A number of different standards are required to realise the interoperable framework. These include:

Manufacturing information standard: In order to provide manufacturing information to the various resources in a CAx chain, it is imperative to store the information in an accessible and robust manner. Figure 4.1 illustrates a categorisation of the manufacturing information relevant to the machining of prismatic parts.

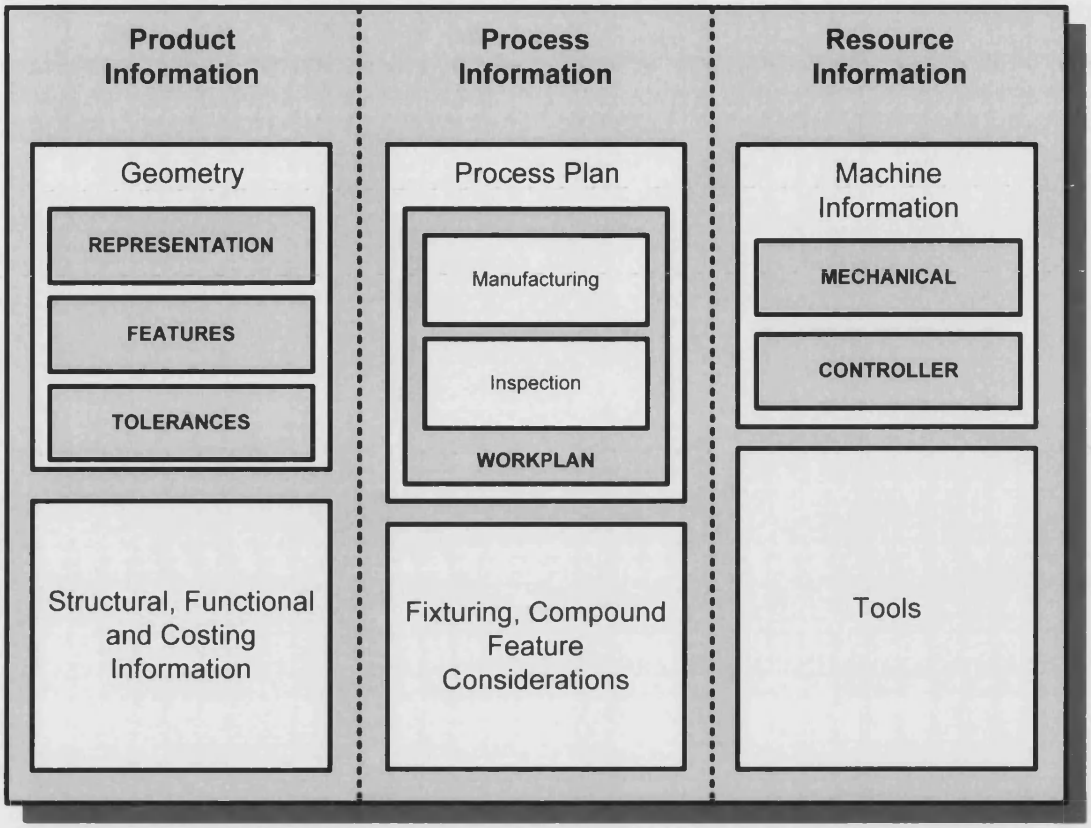


Figure 4.1 - A categorization of the manufacturing information for CNC machining of prismatic parts

It is notable that in order to have full interoperability, process information should be represented in a resource independent manner. STEP-NC provides the data model to represent process plans in such a manner. AP238 and ISO14649 both offer the necessary entities to model a process, but as mentioned in section 3.4 using AP238 carries significant performance penalties without significant advantages over ISO14649. ISO14649 also provides the necessary entities to represent product geometry information. While this representation is not as elaborate as that of STEP-AP224 or STEP-AP203, it is quite adequate for prismatic products and therefore ISO14649 can be considered as appropriate for geometry representation within the context of this research.

At present the author is not aware of any international standards to represent resource information within the context of prismatic part manufacturing. While ISO14649 does provide

cutting tool representations, the mechanical and electronic representations of the machine tools themselves are currently beyond the reach of existing standards. ISO/TC184 has recently started the development of ISO14649-112, a manufacturing resource data model, but at the time of writing of this thesis, the model is not mature enough for use. UML has been used throughout this research to create an object-oriented model to represent machine tools and controllers.

Data transfer standard: A quick survey of recent CNC manufacturing shows that the most popular standard for data transfer is XML. It is however important to note that none of the existing industrial CNC controllers supports XML for data transfer. As a result, ASCII text files have been chosen as the standard for data transfer. STEP-NC files are stored in ISO10303-21 format. Part programs for specific CNC machines are stored in proprietary formats as defined by the vendor.

4.2.2. The Ability to Use Existing Resources without Modification

To ensure functionality of the interoperable CAx framework within an industrial context it is imperative that it interacts with CAx resources without requiring considerable modifications to the resources. The framework should be able to support a variety of resources with a multitude of standards and semantics and adapt accordingly.

4.2.3. Object-Orientation

Object orientation is currently the software development paradigm of choice for most programmers. It provides a reusable and consistent model that is in harmony with the real world that appeals to the human cognition (Booch 1994). The interoperable CAx framework has been designed to be completely object oriented to benefit from the advantages of this philosophy. Java and C++ are the most popular object oriented programming languages (OOP) today. Java provides cross-platform portability or the ability to run the same code on different hardware and software platforms. This feature is useful for implementing the framework on CNC controllers with limited computing capability. Java therefore, has been chosen as the programming language for realisation of the framework.

4.2.4. Semantic Interoperability

In order to ensure the integrity of data exchanges in the information framework the semantics from all resources should be made homogeneous. That is, to make sure that the requester and the provider of the information have the same understanding of the data.

4.3. Framework Functionalities

The goal of the framework is to realise interoperability in a CAD/CAM/CNC chain. It should therefore enable CAx resources to exchange data in a manner that ensures the integrity of the information. This is illustrated in Figure 4.2.

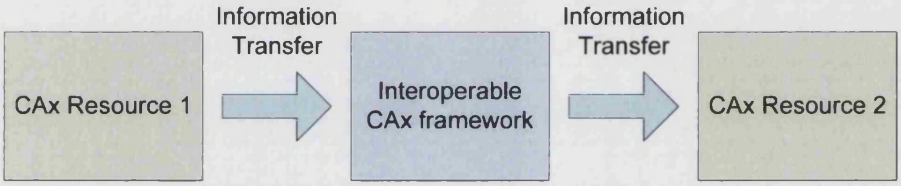


Figure 4.2 - Block diagram of the interoperable CAx framework

The UML sequence diagram in Figure 4.3 illustrates the sequence for transferring manufacturing information from one resource to another and back through the use of the CAx interoperable framework.

In order to minimise human intervention it is necessary to homogenise the semantics of the CAx resources as this eliminates the chance of misinterpretation of data. After this homogenisation, it is possible to store and retrieve the information in a standard format with a neutral syntax without the need for human supervision. Figure 4.4 illustrates this process.

To realise the framework, a data model for standard data storage needs to be defined. Guidelines for semantic interpretation and syntax conversions then need to be established to allow information access through a generic interface with all CAx resources. Finally, means for communicating information from one component to another have to be identified.

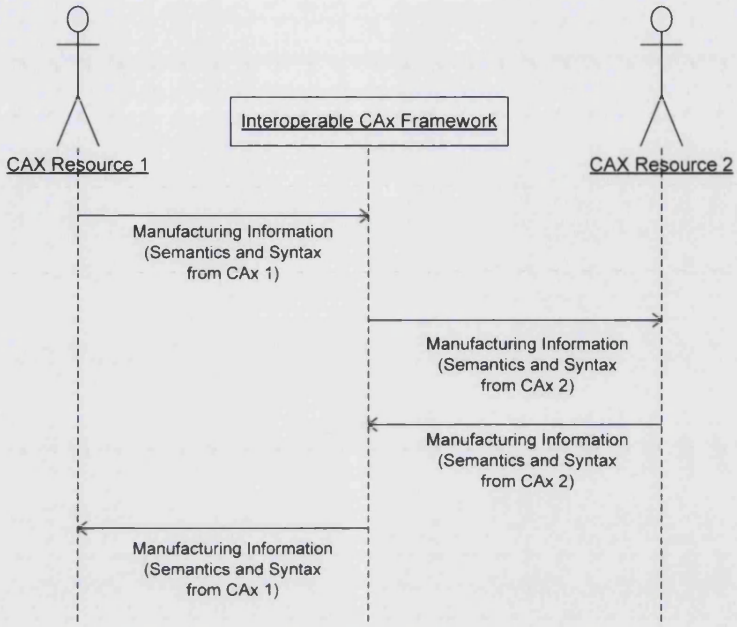


Figure 4.3 - The overall UML sequence diagram for the interoperable CAx framework

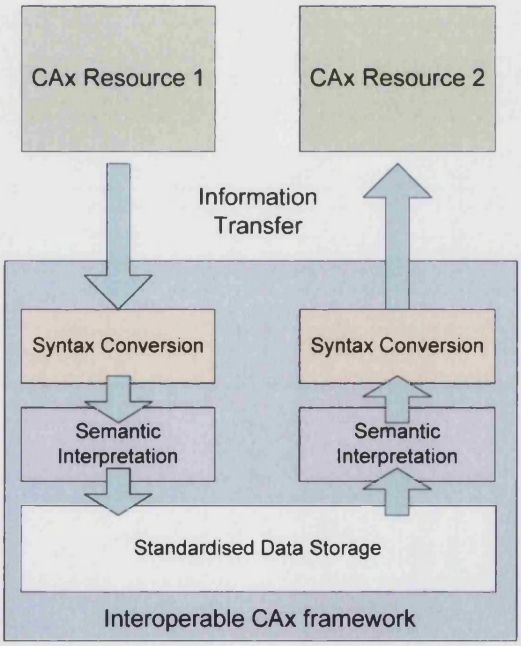


Figure 4.4 - Semantic interpretation and syntax conversion as enablers of interoperability

4.4. Resource Abstraction

Resource abstraction is the process of semantic interpretation combined with syntax translation that provides a generic interface to interact with CAx resources. A generic CNC interface for example would interpret a generic process plan for a product whose geometry is stored in the database and derive the semantics required by a specific CNC controller. The information would then be organised in that controller's syntax and sent to the CNC for machining. Depending on the capabilities of the CNC, the process plan might be presented using a number of feature-based working steps (such as those accessible in a Siemens 840D with ShopMill), a set of pre-defined subroutines (as is the case for a Heidenhain i530TNC) or a G&M Code program suitable for older generation of machines. Figure 4.5 illustrates a comparison between the feature-based ShopMill interface provided by Siemens 840D and the G&M Codes acceptable for an old GE Fanuc controller.

```
E_PO_REC(4,0,0,"9mmSlot","",1,2500.0
,1,3000.0,1,1,2.0,2,2.0,0.0,90,-
12.5,90,2.0,1,7.5,90,20.0,90,35.0,60
.0,5.0,0.0,0.3,0.3,8.0,3.0,0.0,40.0,
0,1);*RO*
E_PO_REC(4,0,0,"9mmSlot","",1,2500.0
,1,3000.0,1,2,2.0,2,2.0,0.0,90,-
12.5,90,2.0,1,7.5,90,20.0,90,35.0,60
.0,5.0,0.0,0.3,0.3,8.0,3.0,0.0,40.0,
0,1);*RO*
```

Bridgeport - Siemens 840D - ShopMill Code

```
T09M06              G01Y75
F2500M03            G03X37.5Y75.5R5
G00X15Y24Z10        G01X12.5
G01Z-12.5           G03X12Y75R5
G01Y76              G01Y25
G01X22              G03X12.5Y24.5R5
G01Y24              G01X37.5
G01X29              G03X38Y25R5
G01Y76              G01Y50
G01X36              G01Z10
G01Y24              G00X-50Y50Z100
G01X38Y25           M00
```

Wadkin – GE Fanuc Code

Figure 4.5 - Comparison of two CNC programs with different levels of information

Chapter 5 presents an object-oriented framework for abstraction of resources, so that regardless of their individual capabilities and particular specifications, all CAx resources can be accessed via a single type of interface.

4.5. Information Models: Creating a Manufacturing Lexicon

In order to ensure semantic interoperability along the CAx chain, it is essential to develop information models corresponding to the categorisation in Figure 4.1. These information models can then be implemented to form a database that can store manufacturing data

regardless of the process stage where the information is generated. Figure 4.6 illustrates the corresponding data models for each category of CNC manufacturing knowledge.

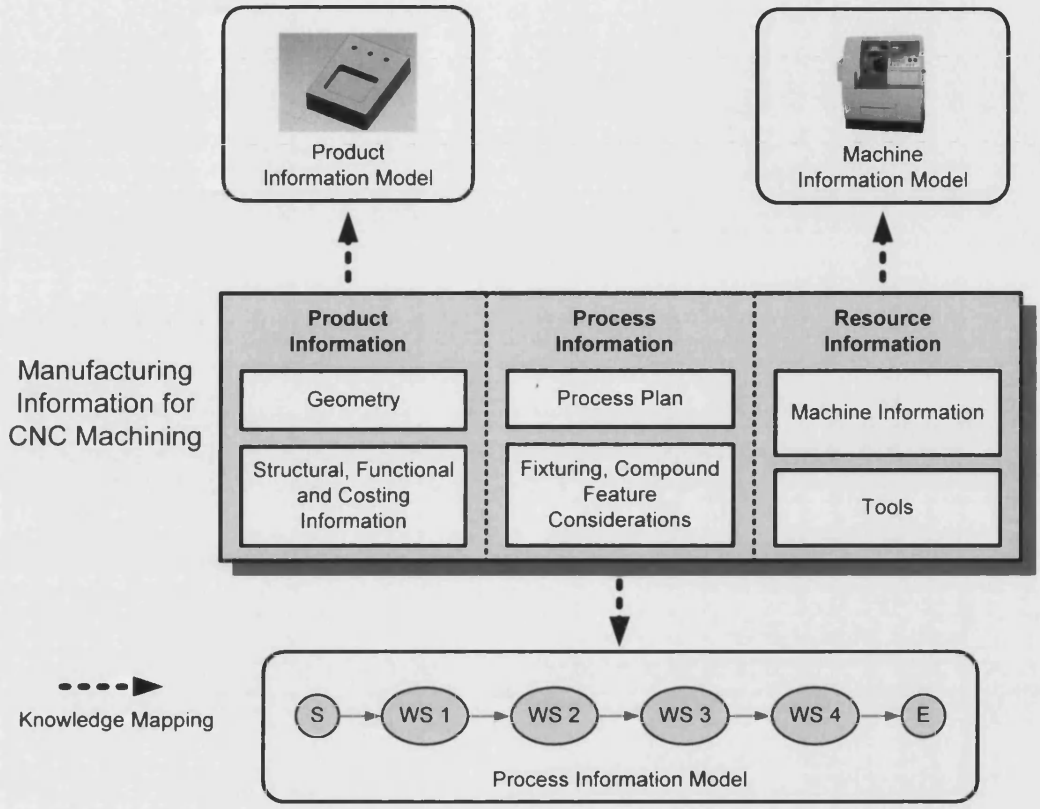


Figure 4.6 - The required information models for the interoperable framework

To ensure that every concept within the domain of prismatic parts manufacturing can be expressed in the database, a comprehensive manufacturing lexicon is needed. According to the literature (Xu and Newman 2006, Saaski 2005, Suh et al. 2006), STEP-NC provides the foundation for such a lexicon for process planning and part geometry information. An example of a process plan represented using entities from the STEP-NC schema is provided in Figure 4.7. In the figure various information models represented in the STEP-NC compliant hierarchy have been identified.

The manufacturing lexicon for the interoperable CAx platform should be able to represent the information about the CAx resources as well as product data and process plans. The literature

indicates that STEP-NC lacks the means to represent resource information models. Furthermore an object-oriented lexicon is required by the interoperability framework, it is therefore necessary to implement the STEP-NC data model in an object-oriented manner and extend it to provide the necessary functionality in storing the complete view of manufacturing information. Chapter 6 presents an innovative approach in achieving this functionality within the context of the framework.

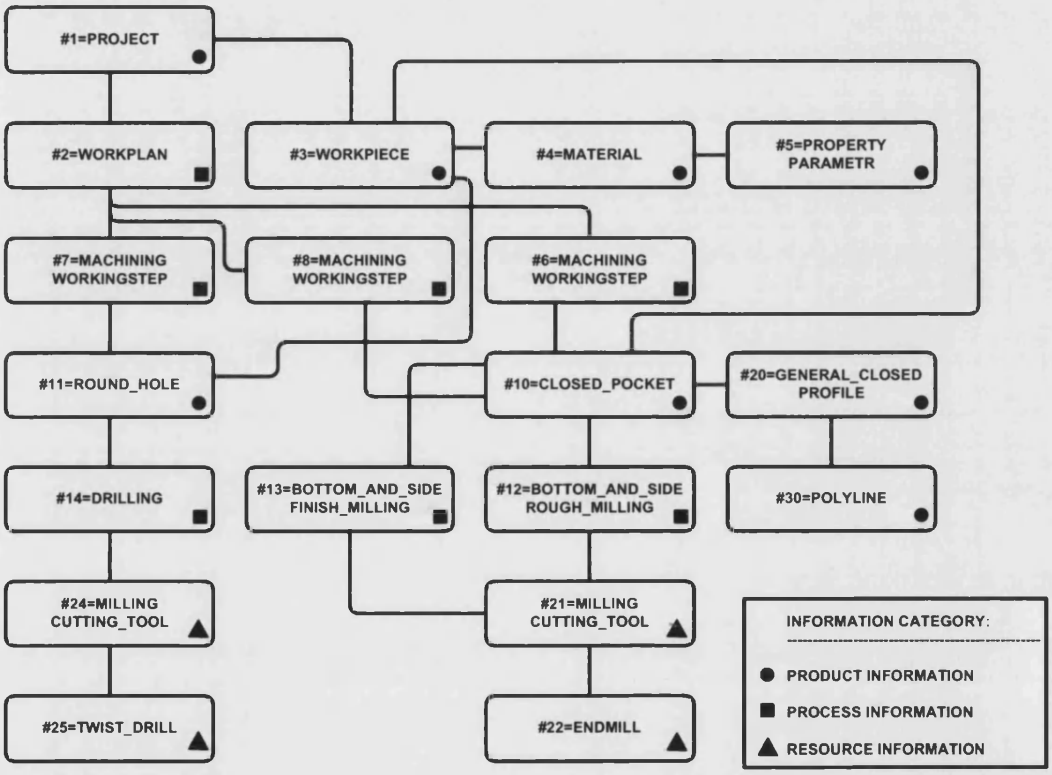


Figure 4.7 - An example of manufacturing information represented using STEP-NC entities

4.6. Information Transfer Mechanisms

The interoperable CAx framework requires a mechanism for transferring the standard encoded manufacturing information to the abstracted resources and vice versa. It is envisioned that the framework will be utilised in a geographically distributed manufacturing network where the various resources are scattered across the globe. The information transfer mechanism will

therefore need to be robust, reliable, efficient and Web enabled to meet the requirements of such a network.

The literature described in 3.4.2 identifies mobile agent systems as a suitable candidate for providing these functionalities, with minimal network requirements for implementation. Chapter 7 investigates the use of mobile agents as the information carriers within the interoperable framework.

4.7. An Overall View of the Interoperable CAx Framework

The functional view of the framework is depicted as an IDEF0 diagram in Figure 4.8. This figure shows how the three mechanisms within the framework provide the necessary tools to enable transfer of information from one CAx resource to another.

The agent based communication system collects data from CAx resource 1 and carries the information to the manufacturing database. The semantics of the information at this stage are those defined by the resource and the syntax is also resource specific. The resource abstraction mechanism is utilised to first standardise the syntax and then interpret the semantics to homogenise them with those contained within the manufacturing lexicon. The resource independent information is then stored within the data models defined in the information encoding and standardisation mechanism.

The data is then retrieved from the manufacturing database and the resource abstraction mechanism is utilised again to homogenise the semantics with those in the second resource and then convert the syntax to a format understandable by the second resource. The agent based communication system then transfers information to the second CAx resource. To transfer information in the opposite direction the same process is employed in reverse. As such any data transfer between two resources is effectively and robustly handled in a universal manner.

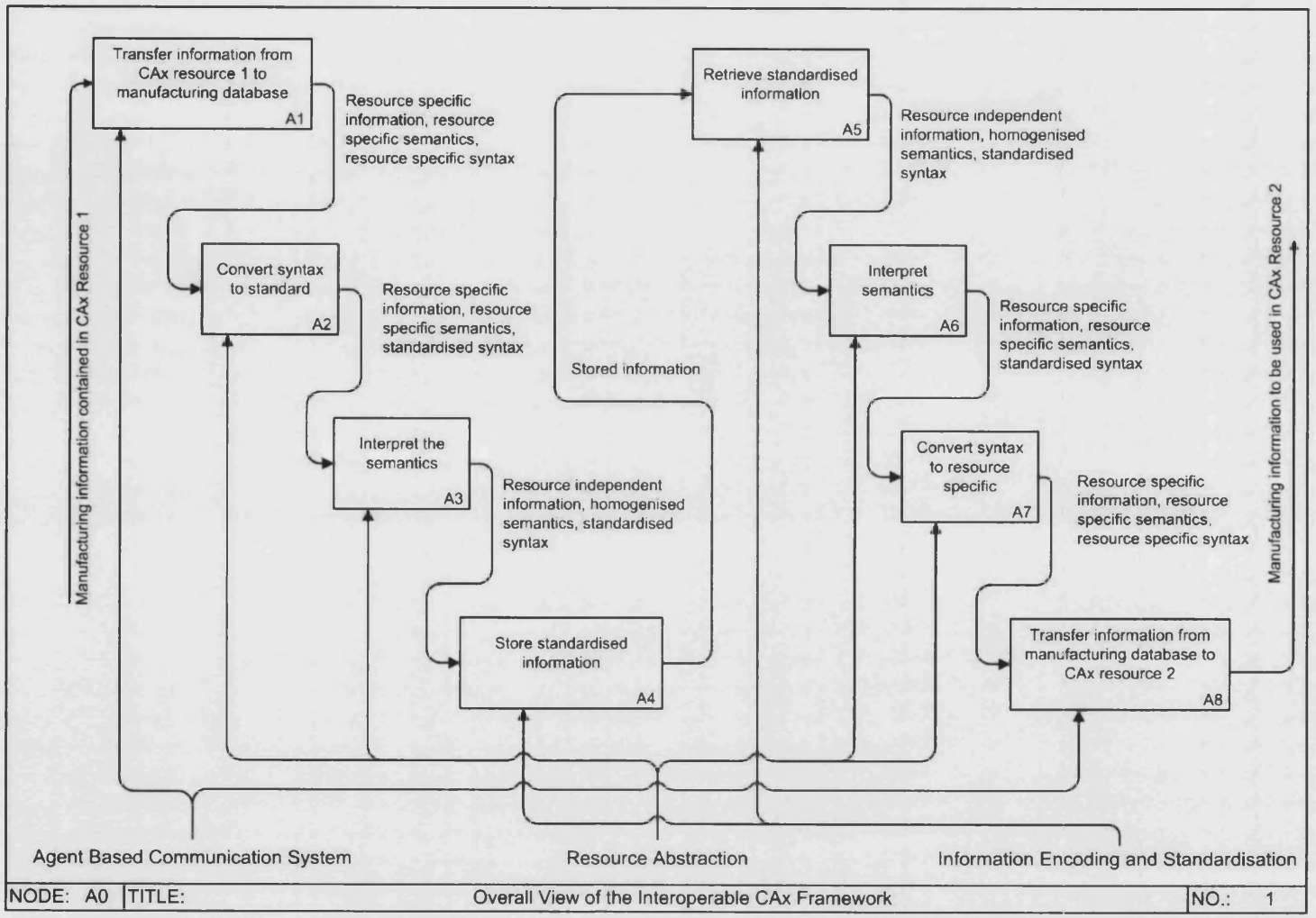


Figure 4.8 - Functional overview of the interoperable CAx framework

Figure 4.9 provides an overall view of the interoperable CAx framework encompassing a number of different CAx resources. Each CAx resource implements the three elements (i.e. resource abstraction, information encoding and communications) of the framework in order to be integrated with the other resources within the framework. In chapters 5, 6 and 7 the various elements of the designed framework will be presented in full detail.

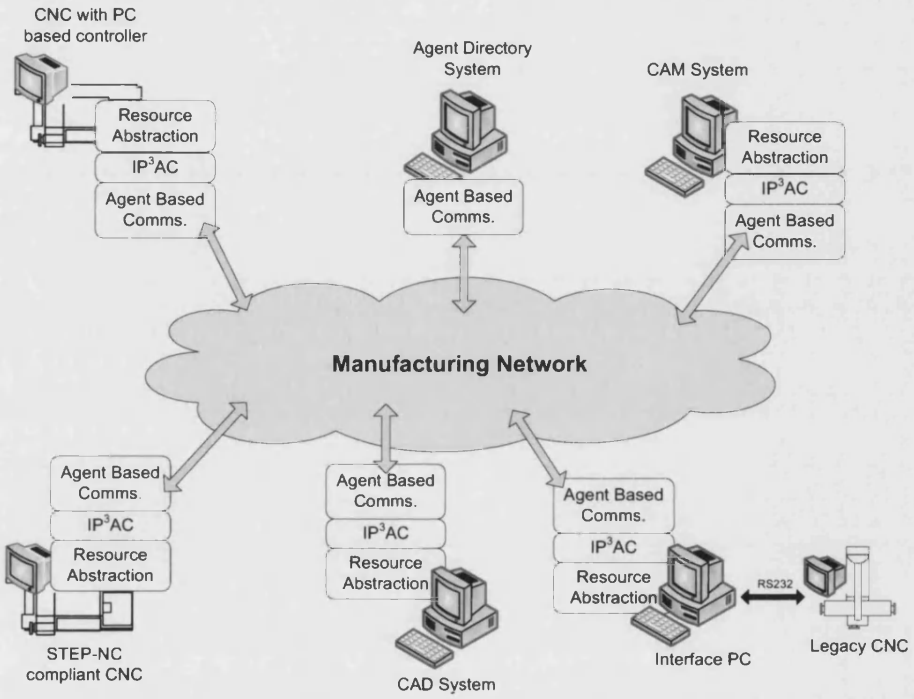


Figure 4.9 - Overall view of the interoperable CAx framework

5. CAx Resource Abstraction

5.1. Introduction

Throughout the CAx chain, manufacturing information is generated, appended and modified. At each stage of the manufacturing process, the data from the previous process is utilised as a foundation to generate the necessary knowledge required by the process in that stage. In order to achieve interoperability, it is imperative to identify this flow of information and study the transformations that the data undergoes until a product is manufactured. This chapter provides an overview of the information flow in the CAx chain for prismatic components. It identifies information transformations and the coupling points between the various data structures, proposes an information homogenisation procedure for manufacturing data. Finally an object oriented resource abstraction framework is constructed. This framework allows manufacturing information to be sent to a CAx resource or requested from a resource without concern for the specific syntax or semantics supported by that resource.

5.2. Information Flow in the Prismatic CAx Chain

5.2.1. Information Flow in the State-Of-The-Art CAx Chain

Figure 5.1 illustrates an example of the typical information flow in the state-of-the-art CAx chain. The geometry of the part is first digitised in a CAD system using entities such as points, lines, curves, boundaries and surfaces. If solid modelling is used within the CAD system then solid volumes are also defined in the CAD system. The information is then conveyed to the CAM system. In some cases, the CAD and CAM systems are unified and therefore the transfer is done seamlessly. Otherwise a standard format is utilised to store the geometry of the part in a file. The file is then read into the CAM system.

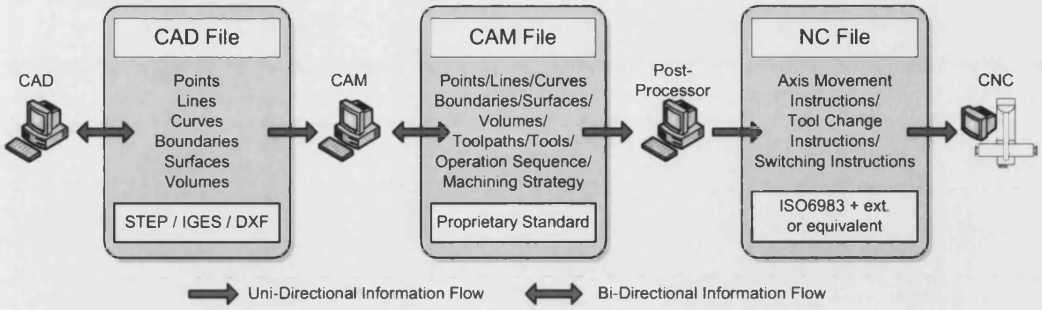


Figure 5.1 - Information flow in state-of-the-art CAx chain

The CAM system interprets the geometry and based on user input generates process plans for manufacturing the component. The process plan which includes machining strategy and operation sequences is then utilised to create tool paths. These paths are defined based on a user selection of cutting tools and the geometry. The CAM file is usually stored in a proprietary vendor specific format. When the process plan is finalised a postprocessor is used to convert the data from product space to machine space and axis movement instructions in the form of G&M Code NC files are generated. As indicated in figure 5.1, this flow is unidirectional and only supports the flow of information downstream to the machine tool.

5.2.2. Information Flow in a STEP-NC Compliant CAx Chain

Through the use of the higher-level STEP-NC standard it is possible to achieve a bi-directional flow of information where the integrity of the data is maintained throughout the CAx chain regardless of modifications and alterations. Figure 5.2 shows the information flow in a STEP-NC enabled CAx chain with a STEP-NC compliant CNC. Even though a commercial STEP-NC CNC does not exist at the time of the writing of this thesis, the ideal bi-directional high-level, high-bandwidth information transfer that could be achieved with such a controller sets the expectations for an interoperable CAx chain. As seen in Figure 5.2, in the STEP-NC compliant chain, the geometry of the component is digitised not only using points, lines, curves, etc. but also geometric and manufacturing features such as cylinders and holes. The standardised information is then passed on to a STEP-NC compliant CAM system where the operation sequences and machining strategy is determined based on user input. These form the basis of a workplan comprising a number of workingsteps to manufacture the component. The

workplan is resource independent; that is, it does not contain resource specific information and every machine can interpret it according to the machine's capabilities.

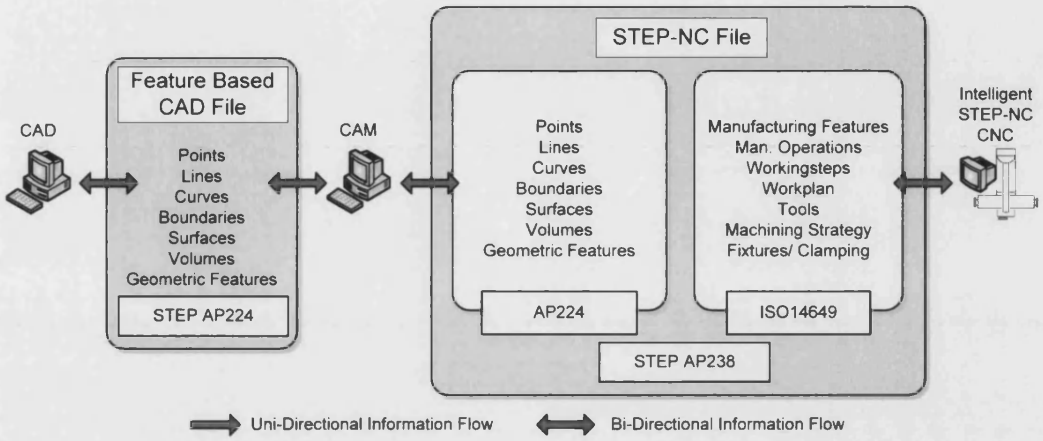


Figure 5.2 - The information flow in a STEP-NC compliant CAx chain

5.3. CAx Resource Abstraction

Each CAx resource in the manufacturing chain accepts data in a specific manner using a specific format. It also exports information in a specific format. These formats and the semantics represented by them are different from one vendor to another. Resource abstraction aims to provide a standard interface where the format of the imported and exported data remains unchanged, regardless of the make and model of a specific resource.

In order to achieve this, the semantics of the information being transferred should be interpreted. When importing data from a CAx resource, first the syntax needs to be standardised. As STEP-NC defines representations for a complete description of data from all CAx resources in prismatic part manufacturing, this translation is mostly syntax conversion. The process is depicted in figure 5.3.

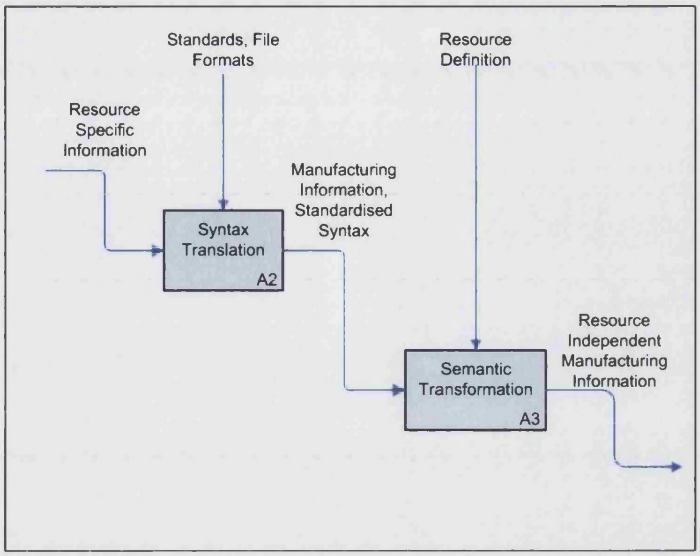


Figure 5.3 - IDEF0 diagram of resource abstraction functionality

Figure 5.4 shows the framework for the CAx resource abstraction for a number of CAx resources namely CAM systems and CNCs. The CAx system definition is used as the basis to translate syntaxes and transform and homogenise the semantics to represent the information with the manufacturing Interlingua.

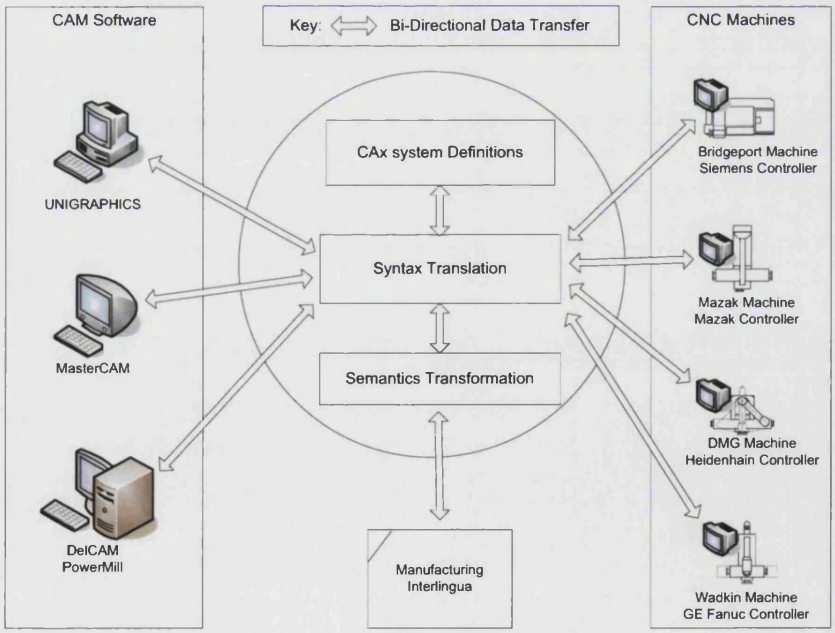


Figure 5.4 - CAx resource abstraction

In effect resource abstraction allows a CAx resource with proprietary information interfaces to be packaged inside an entity with standardised interfaces as pictured in figure 5.5.

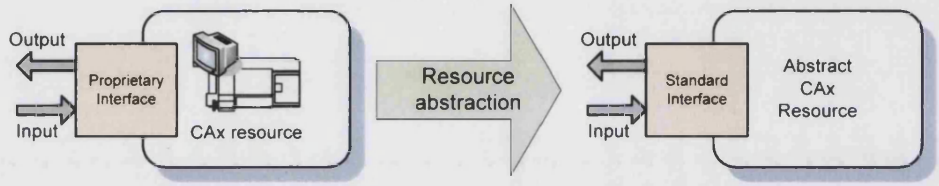


Figure 5.5 - Abstracted resource interface

In the object-oriented interoperable framework, this entity can be modelled as an object. The object class should define interfaces for methods to exchange standardised information. A direct implementation of the object oriented interface will require the intended CAx resource to offer a certain degree of programmability (namely the ability to run a Java virtual machine) and direct access to its low-level components. The semantics from the data sent to the object is interpreted and translated to the syntax required for low-level components. The feedback is taken from the low-level components and transferred in a similar manner through the standard interface. An example of this type of implementation on a CNC controller can be seen in figure 5.6.

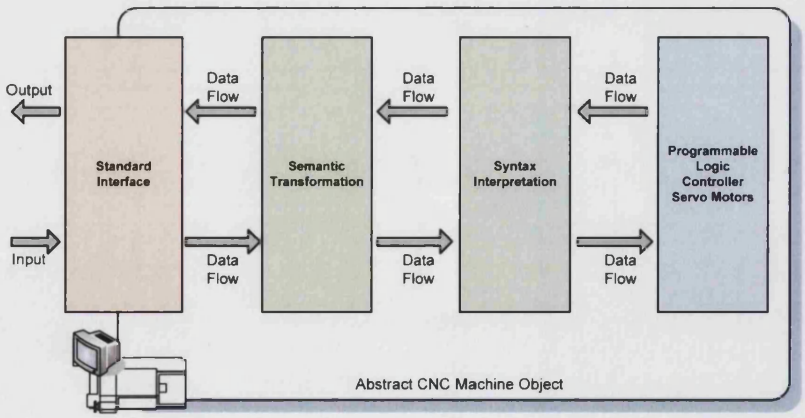


Figure 5.6 - A CNC machine with direct implementation of resource abstraction

In chapter 8 a CAD/CAM system based on the direct implementation of the abstraction interface is developed using the Java language. The CAD/CAM program in chapter 8 is a prototype for integration of future generations of STEP-NC compliant software within the interoperable CAx framework.

While this type implementation might seem desirable, as most CAx resources do not offer the necessary degree of programmability in order to conform to the requirement defined in 4.2.2, it is essential to specify an indirect implementation approach. This approach could be used for closed CAx resources that do not offer user programmability. Some resources like CAD and CAM systems might not support complete programmability to protect intellectual property while CNC controllers might simply not have enough computational power. The indirect implementation approach in figure 5.7 can be utilised to abstract these types of resources.

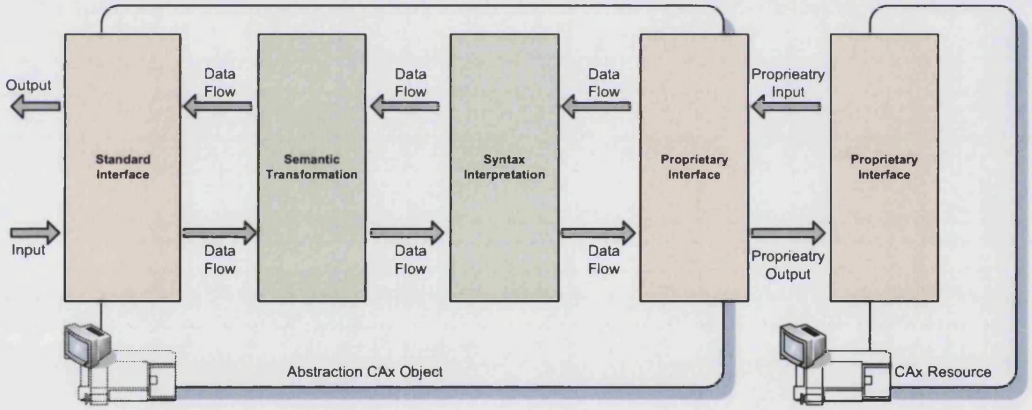


Figure 5.7 - Indirect implementation of CAx resource abstraction

In the indirect approach, the standard information is transformed semantically to match the semantics of the CAx resource and formatted according to its syntax. It is then sent to the resource in the proprietary format. The feedback is gathered from the resource in the proprietary format and translated into standard syntax and after homogenisation of semantics is transferred back to the standard interface of the abstraction object. In chapter 8 two CNC controllers are abstracted using the indirect approach. Information transfer in both directions is defined for one controller while a uni-directional abstraction is realised for the other.

In the event that CAx resources that directly implement the manufacturing Interlingua (i.e. STEP-NC controllers or STEP-NC compliant CAD/CAM systems) become commercially popular, the interoperable framework can directly interface with such resources with no additional steps being required. As most of the current CAx resources are of the latter type and require the implementation of the abstraction object as a separate entity, this type of abstraction is commercially essential and is a major advantage of the interoperable framework over previous solutions.

5.4. CAX Resource Abstractor Object

The four interfaces that the abstractor object will need to implement have been identified as follows: a standard input interface that allows the object to receive standardised manufacturing information, a standard output interface to provide the standardised feedback, a proprietary output interface to write the resource specific information and a proprietary input interface to read the resource specific information. In prismatic part manufacturing three types of resources are defined: CAD, CAM and CNC corresponding to the functions in figure 5.1. The UML class diagram showing the CAx resource abstractor class and its subclasses is illustrated in figure 5.8.

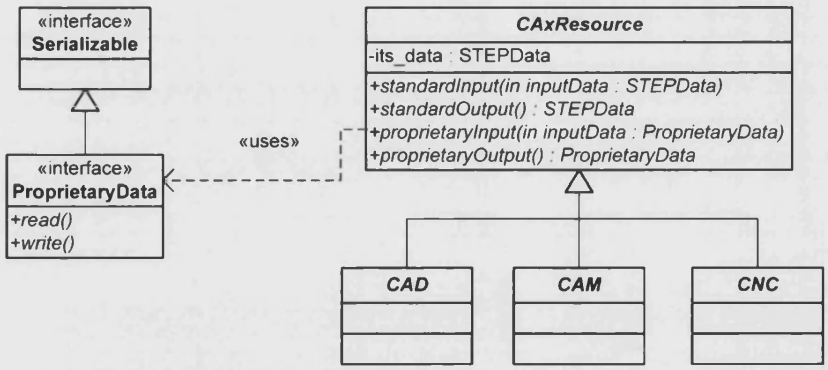


Figure 5.8 - UML diagram of the resource abstractor class

In order for each resource in the manufacturing chain to be able to communicate with the interoperable framework, a class, extending one of the above classes has to be defined with the four data transfer methods implemented. For example the classes required for defining the Siemens 840D controller and an old GE Fanuc controller are shown in figure 5.9.

While the external interfaces of the two classes are identical the implementation of the proprietary input and output methods are substantially different.

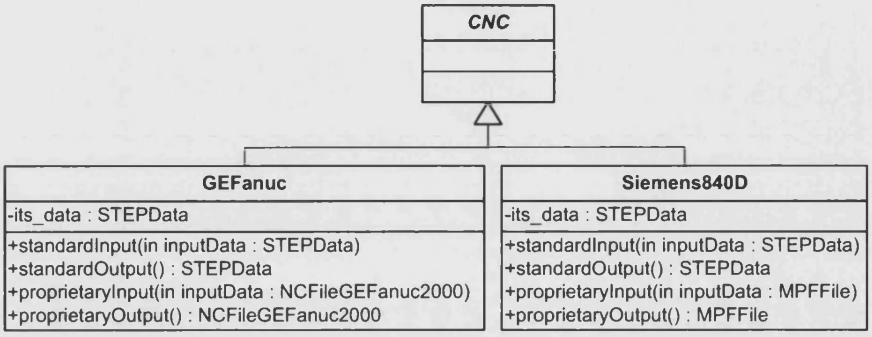


Figure 5.9 - The resource abstractor classes for Siemens 840D and GE Fanuc controllers

The Siemens controller with its ShopMill shop floor programming system is capable of handling manufacturing features such as holes, pockets and generic surfaces. The GE controller is a basic G&M code controller. As a result the resulting proprietary files are considerably different. Figure 5.10 shows a simple component with its STEP-NC data in figure 5.11.

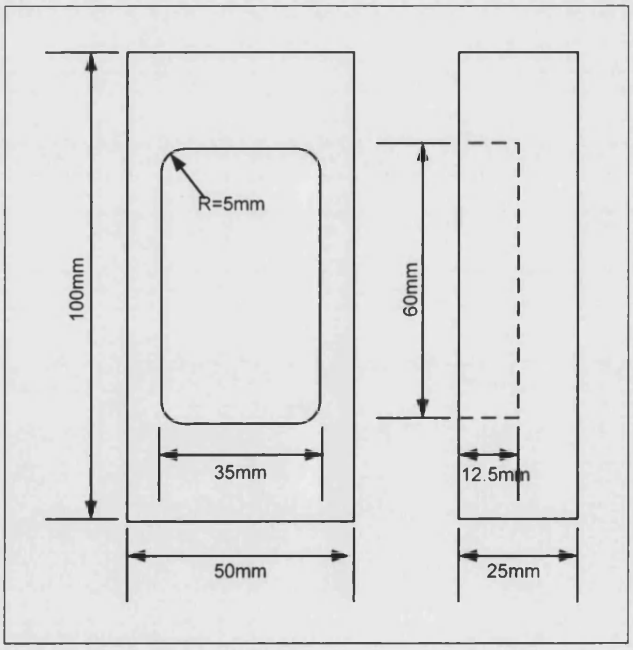


Figure 5.10 - A simple prismatic component

```
ISO-10303-21;
 HEADER;
 FILE_DESCRIPTION(('ISO 14649-11 FILE','AUTOMATIC OUTPUT GENERATED BY INTEROPERABLE FRAMEWORK'),'1');
FILE_NAME('EXAMPLE.STP','2007-05-07',('AYDIN NASSEHI','STEPHEN NEWMAN'),('BATH UNIVERSITY'),$,'ISO 14649',$);
FILE_SCHEMA(('combined_schema'));
ENDSEC;

DATA;
#1=PROJECT('Test Pocket 1',#2,(#3),$,$,$);
#2=WORKPLAN('Main Workplan',(#4,#5),$,$,$);
#3=WORKPIECE('Simple Workpiece',#20,0.01,$,$,#21,());
#4=MACHINING_WORKINGSTEP('Milling working step for roughing pocket',#6,#7,#8,$);
#5=MACHINING_WORKINGSTEP('Milling working step for finishing pocket',#6,#7,#13,$);
#6=PLANE('Security plane for pocket',#9);
#7=CLOSED_POCKET('pocket',#3,(#8,#13),#14,#15,(),$,#16,#17,#18,#19);
#8=BOTTOM_AND_SIDE_ROUGH_MILLING($,$,'Rough milling for pocket',$,$,$,$,$,$,$,$,$,$,$,$);
#9=AXIS2_PLACEMENT_3D('Axis for Security axis placement',#10,#11,#12);
#10=CARTESIAN_POINT('point for Security axis placement',(0.0,0.0,10.0));
#11=DIRECTION('axis direction for Security axis placement',(0.0,0.0,1.0));
#12=DIRECTION('reference direction for Security axis placement',(1.0,0.0,0.0));
#13=BOTTOM_AND_SIDE_FINISH_MILLING($,$,'Rough milling for pocket',$,$,$,$,$,$,$,$,$,$,$,$);
#14=AXIS2_PLACEMENT_3D('Axis for pocket placement',#27,#28,#29);
#15=PLANE('Pocket Depth plane',#30);
#16=PLANAR_POCKET_BOTTOM_CONDITION();
#17=TOLERANCED_LENGTH_MEASURE(5.0,$);
#18=TOLERANCED_LENGTH_MEASURE(0.5,$);
#19=GENERAL_CLOSED_PROFILE(#34,#35);
#20=MATERIAL('Test Material','TST-1',(#22));
#21=BLOCK('block for Simple Workpiece',#23,50.0,100.0,25.0);
#22=NUMERIC_PARAMETER('E',2100000.0,'Kg/m4');
#23=AXIS2_PLACEMENT_3D('Axis for axis for block for Simple Workpiece',#24,#25,#26);
#24=CARTESIAN_POINT('point for axis for block for Simple Workpiece',(0.0,0.0,-25.0));
#25=DIRECTION('axis direction for axis for block for Simple Workpiece',(0.0,0.0,1.0));
#26=DIRECTION('reference direction for axis for block for Simple Workpiece',(1.0,0.0,0.0));
#27=CARTESIAN_POINT('point for pocket placement',(0.0,0.0,0.0));
#28=DIRECTION('axis direction for pocket placement',(0.0,0.0,1.0));
#29=DIRECTION('reference direction for pocket placement',(1.0,0.0,0.0));
#30=AXIS2_PLACEMENT_3D('Axis for Depth placement',#31,#32,#33);
#31=CARTESIAN_POINT('point for Depth placement',(0.0,0.0,-12.5));
#32=DIRECTION('axis direction for Depth placement',(0.0,0.0,1.0));
#33=DIRECTION('reference direction for Depth placement',(1.0,0.0,0.0));
#34=AXIS2_PLACEMENT_3D('Axis for profile placement',#36,#37,#38);
#35=POLYLINE('Pocket boundary',(#39,#40,#41,#42,#43));
#36=CARTESIAN_POINT('point for profile placement',(0.0,0.0,0.0));
#37=DIRECTION('axis direction for profile placement',(0.0,0.0,1.0));
#38=DIRECTION('reference direction for profile placement',(1.0,0.0,0.0));
#39=CARTESIAN_POINT('Point 1 for pocket polyline',(7.5,20.0,0.0));
#40=CARTESIAN_POINT('Point 2 for pocket polyline',(42.5,20.0,0.0));
#41=CARTESIAN_POINT('Point 3 for pocket polyline',(42.5,80.0,0.0));
#42=CARTESIAN_POINT('Point 4 for pocket polyline',(7.5,80.0,0.0));
#43=CARTESIAN_POINT('Point 5 for pocket polyline',(7.5,20.0,0.0));
ENDSEC;
END-ISO-10303-21;
```

Figure 5.11 - STEP-NC process plan for the simple prismatic component

The standard input methods of abstractors for both controllers read the same file, but when the proprietary output method is invoked the Siemens abstractor generates the code in figure 5.12 and the GE abstractor generates the code in figure 5.13.

```
E_HEAD(268439551,0.0,0.0,-25.0,50.0,100.0,25.0,71.0,17.0,25.0,25.0,0.0,0.0,6.0);*RO*
E_PO_REC(4,0,0,"9mmSlot","",1,2500.0,1,3000.0,1,1,2.0,2,2.0,0.0,90,-
12.5,90,2.0,1,7.5,90,20.0,90,35.0,60.0,5.0,0.0,0.3,0.3,8.0,3.0,0.0,40.0,0,1);*RO*
E_PO_REC(4,0,0,"9mmSlot","",1,2500.0,1,3000.0,1,2,2.0,2,2.0,0.0,90,-
12.5,90,2.0,1,7.5,90,20.0,90,35.0,60.0,5.0,0.0,0.3,0.3,8.0,3.0,0.0,40.0,0,1);*RO*
```

Figure 5.12 - Output generated by the proprietary output method of the Siemens 840D abstractor object

```
T09M06              G01Y75
F2500M03            G03X37.5Y75.5R5
G00X15Y24Z10        G01X12.5
G01Z-12.5           G03X12Y75R5
G01Y76              G01Y25
G01X22              G03X12.5Y24.5R5
G01Y24              G01X37.5
G01X29              G03X38Y25R5
G01Y76              G01Y50
G01X36              G01Z10
G01Y24              G00X-50Y50Z100
G01X38Y25           M00
```

Figure 5.13 - Output generated by the proprietary output method of the GE Fanuc abstractor object

Bi-directional information transfer can be easily achieved in high-level feature-based controllers such as the Siemens 840D. The semantics contained within the controller understandable feature format and the feature format in the interoperable framework are the same.

With low-level G&M controllers such as the GE Fanuc the issue is somewhat different. Here the machine understandable format no longer contains the feature or process information. Fundamental changes to the low-level program will therefore have severe effects on the semantics contained within the feature-based system. Further information in the form of a CAD file or a CAM file is required in such systems for translating proprietary input into standardised output. For a detailed study refer to Shin et al. (2007).

5.5. External Definition of CAx Resources

Instead of coding the implementation of CAx resources in individual classes, it is possible to employ an open standard such as XML to describe the CAx resource. With this approach a single implementation can be utilised for abstracting an entire hierarchy of resources such as CNC controllers.

The information provided in the XML definition of resources varies in complexity. Details ranging from machine serial number, model and make to actual implementation of different manufacturing processes could be included in the machine description. XML has the versatility of providing the functionality without adding to the system complexity. Figures

5.14 and 5.15 show simplified definitions of the Siemens 840D and GE Fanuc controllers mounted on two machine tools defined in XML files.

```
- <machinedef>
  - <information>
      <id>b840d1_aus_12</id>
      <make>Brdigeport</make>
      <model>VMC 1000K</model>
      <maxaxis>5</maxaxis>
      <availableaxis>3</availableaxis>
      <maxspindlerpm>8000</maxspindlerpm>
    </information>
  - <controller>
      <make>Siemens</make>
      <model>840d</model>
      <omit_undefined_parameters>true</omit_undefined_parameters>
    - <capabilities>
      + <Rapid>
      - <TwoDlineinterpolation>
          <capable>true</capable>
          <syntax>G01 X@axis0 Y@axis1 Z@axis2 A@axis3 B@axis4 F@feedrate;</syntax>
        + <parameters>
        </TwoDlineinterpolation>
      + <TwoDarcradiusinterpolation>
      - <TwoDrectangularpocket>
          <capable>true</capable>
          <syntax>E_PO_Rec(@x0,@y0,@x1,@y1,@planaradius,@feedrate);</syntax>
        - <parameters>
          - <optional>
              <prm>@feedrate</prm>
            </optional>
          - <required>
              <prm>@x0</prm>
              <prm>@y0</prm>
              <prm>@x1</prm>
              <prm>@y1</prm>
              <prm>@planaradius</prm>
            </required>
          </parameters>
        </TwoDrectangularpocket>
      </capabilities>
    </controller>
  + <dimensions>
  </machinedef>
```

Figure 5.14 - XML definition of Siemens 840D controller on a Bridgeport machine

The unified implementation of semantic transformation and syntax translation aims to convert the standardised manufacturing data into the highest level of manufacturing representations supported by the resource. This allows any resource specific optimisations to be utilised by the CAx chain. For example when manufacturing the simple part illustrated in figure 5.10, if the resource is capable of representing a rectangular pocket with one instruction, the two dimensional rectangular pocket will be translated to a rectangular pocket manufacturing instruction.

```
- <machinedef>
  - <information>
      <id>wfanuc_hk_11</id>
      <make>Wadkin</make>
      <model>V5 10</model>
      <maxaxis>3</maxaxis>
      <availableaxis>2.5</availableaxis>
      <maxspindlerpm>5000</maxspindlerpm>
    </information>
  - <controller>
      <make>GE Fanuc</make>
      <model>2000</model>
      <omit_undefined_parameters>true</omit_undefined_parameters>
    - <capabilities>
      + <Rapid>
      - <TwoDlineinterpolation>
          <capable>true</capable>
          <syntax>@l G01 X@axis0 Y@axis1 Z@axis2 A@axis3 B@axis4 @feedrate</syntax>
        - <parameters>
          - <optional>
              <prm>@axis0</prm>
              <prm>@axis1</prm>
              <prm>@axis2</prm>
              <prm>@axis3</prm>
              <prm>@axis4</prm>
              <prm>@feedrate</prm>
            </optional>
          - <required>
              <prm>@l</prm>
            </required>
          </parameters>
        </TwoDlineinterpolation>
      + <TwoDarcradiusinterpolation>
      - <TwoDrectangularpocket>
          <capable>false</capable>
        </TwoDrectangularpocket>
      </capabilities>
    </controller>
  + <dimensions>
  </machinedef>
```

Figure 5.15 - XML definition of GE Fanuc controller on a Wadkin machine

In the case that the resource does not have the capability of handling two dimensional pockets, the semantic translator tries to combine two simpler processes to create the pocket. These processes are the clear rectangular area process and the two dimensional contouring process. If the resource is incapable of comprehending this level of commands, the pocket will be translated into a collection of linear interpolations and circular interpolations. The flowchart for the implementation of this unified proprietary output generation method for a simple rectangular pocket is shown in Figure 5.16

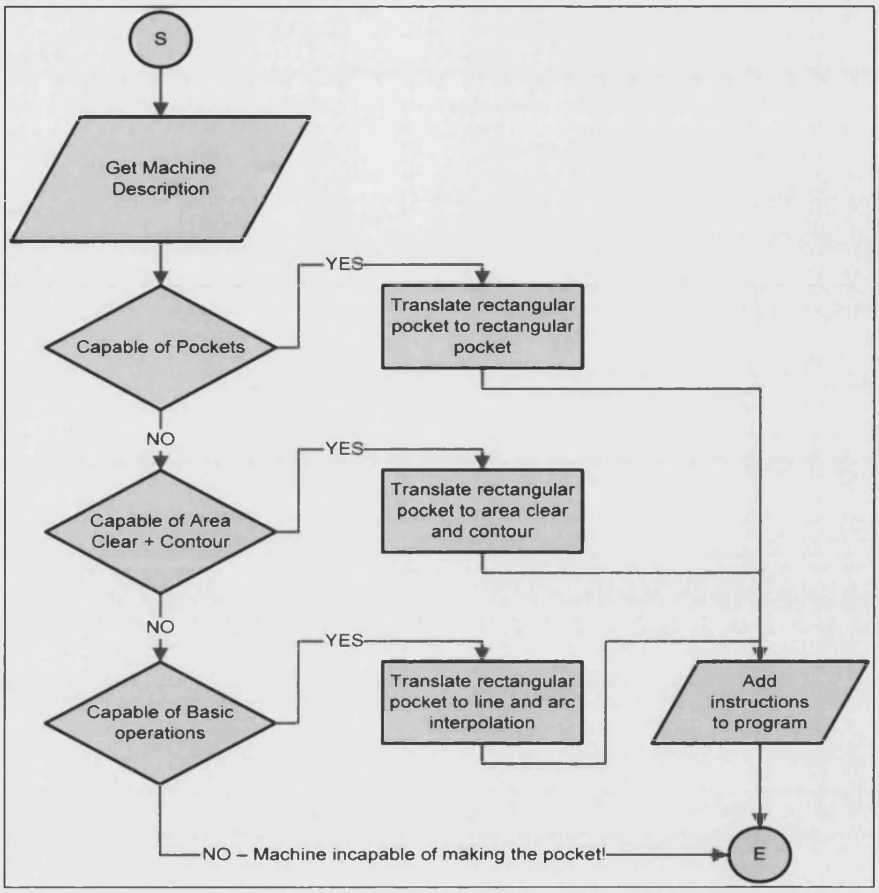


Figure 5.16 - Resource abstraction for a simple process plan

5.6. Summary

In this chapter an approach for abstraction of CAx resources was proposed. The approach utilises XML definitions of various elements of the CAx specific data syntax and links the semantics contained within these elements with the lexicon of the manufacturing Interlingua. This allows all CAx resources to be accessed via standardised interfaces and enables semantic homogenisation of data.

6. A Computational Platform for a Comprehensive CAx Manufacturing Lexicon

6.1. Introduction

As mentioned in chapters 4 and 5, an essential requirement for enabling interoperability in the CAx manufacturing chain is a comprehensive manufacturing lexicon. This lexicon will allow the information generated along the CAx chain to be recorded using a neutral language that can be transformed and translated for all resources. In order for the various resources to be able to store and manipulate information using the lexicon as the semantic context, a computational implementation is required. As mentioned in chapter 5, STEP-NC provides a suitable foundation for constructing such a lexicon and an object-oriented computational platform based on STEP-NC can support the interoperable CAx system. This chapter identifies the various layers of manufacturing information that need to be stored in the interoperable framework to examine the suitability of STEP-NC for representing the information. An object-oriented computational platform based on STEP-NC called the integrated platform for process planning and control (IP^3AC) is then specified, designed and realised.

6.2. Layers of Manufacturing Information

In a CAx chain the starting point for the generation of information is always in the definition of geometry. The final information generated is the electrical signals that drive the axis motors in the CNC machine tools. In the transformation process of the evolving data from design to axis movement instructions, a body of knowledge and information is generated. These items of information can be logically organised into a number of layers as illustrated in Figure 6.1.

Geometry information is product oriented and does not include information relevant to the manufacturing processes. The notable exception is that when a component is defined using manufacturing features (as is the case in STEP-AP224), the geometry is inherently linked with the manufacturing processes. In other cases, the link between the geometry and the process plan is maintained by CAx software and is not reflected in the data structures.

Workingsteps and operations contain generic and non-resource specific information about the sequence of operations and manufacturing strategies that are necessary to machine the desired part. In the state-of-the-art, this information is stored in the CAM system and is not always accessible outside the boundaries of the CAM software itself.

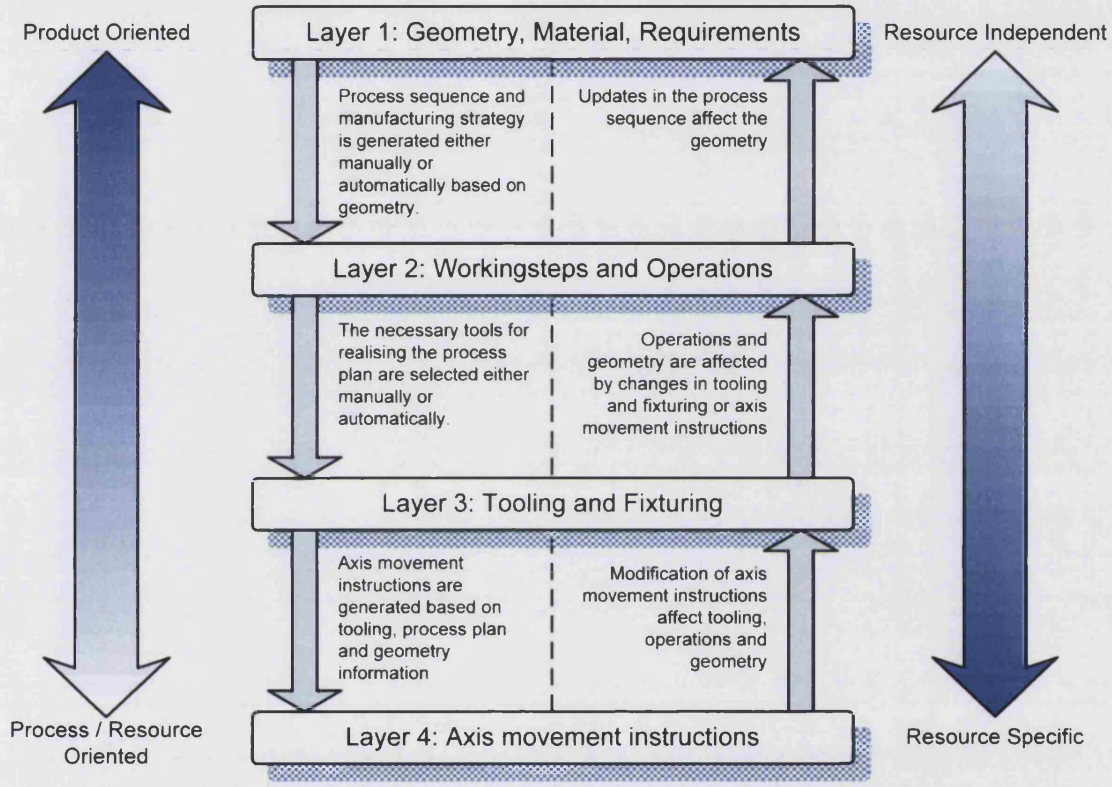


Figure 6.1 - Information layers required for CNC manufacturing

It should be noted that completely generic process plans are impossible to define unless data structures to store multi-route flexible process plans are constructed. For example machining a part on a 5-axis milling machine might require a considerably different process plan to the one that is required for machining the same part in multiple set-ups on a 3-axis milling machine. As the scope of this research however, is focused on prismatic components, generic process plans contained in linear data structures are feasible in most cases.

At the lower layer of tooling and fixturing, the information starts to become resource and process centric instead of product centric and as such, resource specific information is

required. Tool shapes, process capabilities and fixturing requirements for producing a component are just a few items of information that are required at this level to support CNC machining. In the state-of-the-art some of this knowledge is stored in computer programmes while most of it is only known by the operator's themselves.

At the lowest level, manufacturing information is expressed using axis movement instructions or electric signals that are sent to motors attached to the various axis of the machine tool. Currently this information is created by the postprocessor in the form of G&M codes. In a STEP-NC compliant CAx chain, this information is generated at runtime on the controller itself when interpreting the STEP-NC program data.

6.3. STEP-NC, the Interlingua for Interoperability

In order to abstract CAx resources it is necessary to choose a neutral form of expression to serve as the Interlingua to convey generic information. A language should be capable of representing the semantics required by the four layers of information identified in figure 6.1 to be able to serve as this Interlingua. The choice of the target language is essential because it should be expressive enough in order to represent explicit and precise knowledge. On the other hand, it is important to compromise between expressiveness and complexity. (Da Silva 2006)

STEP-NC provides the blueprints for the necessary entities to represent information from all four layers as defined in figure 6.1. Some of these entities lack the sophistication required for representing complex parts that require 5-axis machining or combined turn/mill machine tools. Nevertheless STEP-NC's data structures' capability for storing information for prismatic parts is more than adequate.

Table 6.1 lists a number of key entities from the ISO14649 STEP-NC schema and the respective layer for each entity.

Table 6.1 - Key entities in ISO14649 and their respective information layer

Entity Name	Entity Description	Information Layer
Project	Top level entity in the ISO14649 data model. Every other item of information is linked to this entity.	All four
Workpiece	Describes the workpiece that is used to manufacture the component	Layer 1
Manufacturing_Feature (with its subtypes: region, two5D_manufacturing_feature, machining_feature, planar_face, pocket, slot, step, etc.)	Describes the manufacturing features that comprise the part.	Layer 1
Executable, Workingstep, Operation, machining workingstep, workplan	Describe the stages of the process plan	Layer 2
Machining_Tool and its subclasses	Describe the tool capabilities and geomtery	Layer 3
Workpiece_setup	Can be used to define fixturing	Layer 3
Toolpath_list	Describe axis movement instructions	Layer 4

As seen in table 6.1, ISO14649 offers the data structures to adequately represent information from all four layers in prismatic part manufacturing. Geometry, operations, tools and axes movements can all be represented using the elements in the STEP-NC data model.

6.4. The Integrated Platform for Process Planning and Control (IP^3AC)

The STEP standard offers SDAI (see section 3.3.2) as the formal method of manipulating a population of STEP data. Due to the various problems identified in section 3.3.2 in an object oriented implementation of SDAI, a new platform titled IP^3AC has been developed as a replacement. The platform provides encapsulation of STEP-NC data in objects and by defining collections of objects allows a Java program to manipulate STEP-NC manufacturing information in native Java data structures. Figure 6.2 shows an example of data structures employed to represent the manufacturing data contained within an ISO14649 text file.

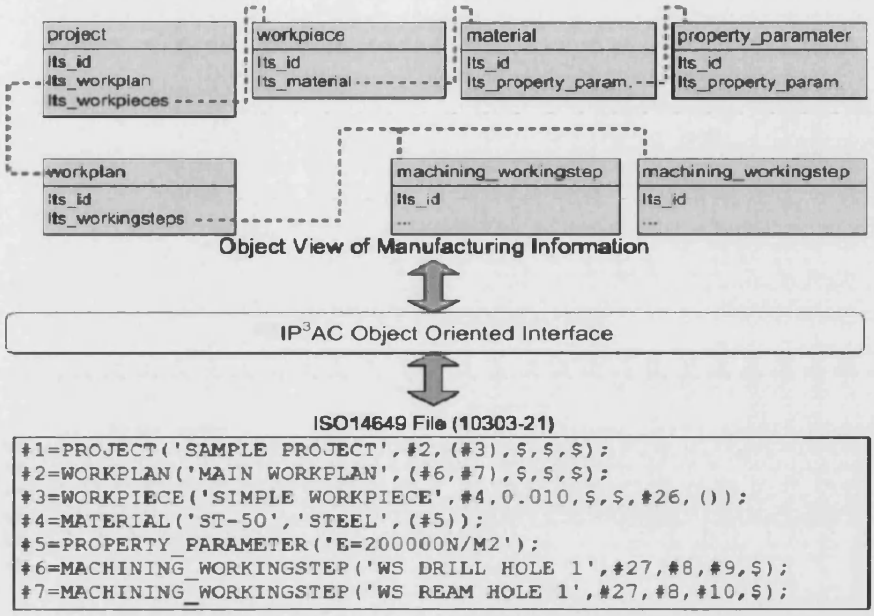


Figure 6.2 - Object oriented encapsulation of manufacturing information in IP[3]AC

To achieve seamless integration of the application with the data structures, it is necessary to generate early bindings for the EXPRESS entities contained within an ISO14649 data model by translating the entity definitions into Java class definitions. Considering that EXPRESS entities in ISO14649 are defined in an object-oriented like manner the translation does not need to substantially change the semantics involved in the creation of the models; in most cases it is only a matter of translating the data models' syntax. It is important to note that EXPRESS only defines states and no behaviours and therefore no methods can be derived directly from the data structures contained within an EXPRESS schema.

One important structural difference between EXPRESS and Java is that EXPRESS supports multiple inheritances but Java only supports single inheritance. Figure 6.3 shows a sample of an entity in ISO14649 where multiple inheritances have been defined.

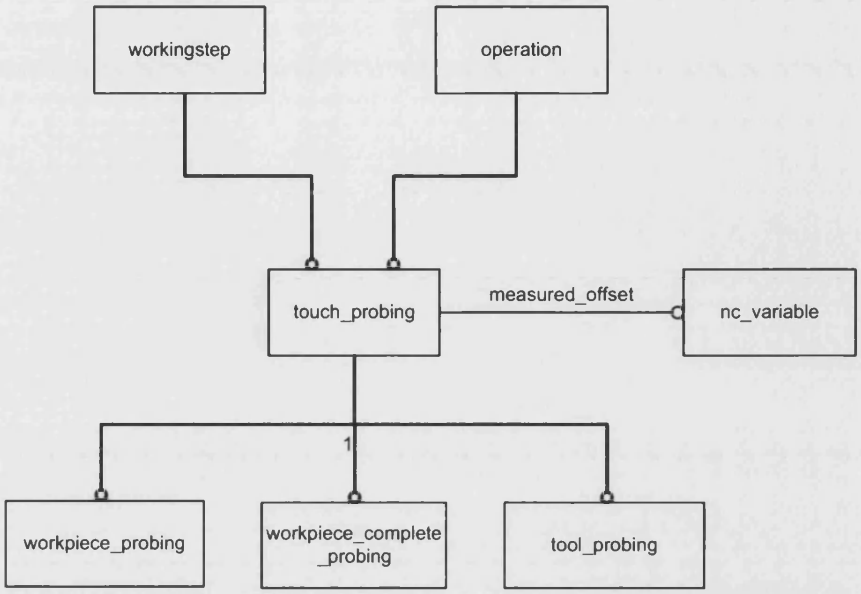


Figure 6.3 - An example of multiple inheritances in ISO14649

In order to define the same structure using Java constructs, interfaces for each class need to be defined. This is shown in Figure 6.4.

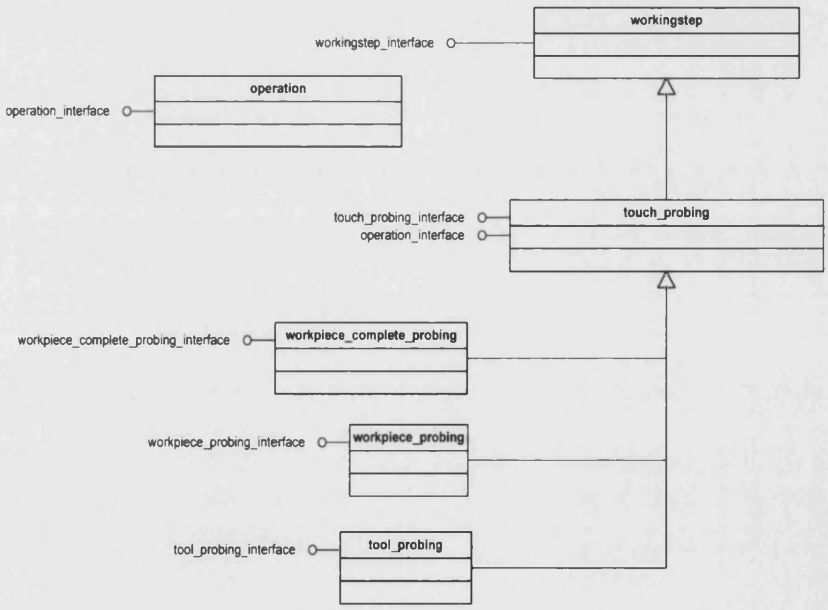


Figure 6.4 - Defining EXPRESS's multiple inheritances using Java interfaces

By defining interfaces for each class it has been possible to achieve the same information structure represented within the STEP-NC model, without requiring multiple inheritances. It is also noteworthy that multiple inheritances are usually considered to be a sign of bad design by programmers today (Garrido, 2003) and in the future versions of the standard, removing these links should be taken under consideration by the respective committees.

In order to translate the EXPRESS schema into a Java class collection, it has been necessary to consider the various elements of the schema as follows:

6.4.1. Entities

The definition of entities in IP^{3}AC begins by creating Java interfaces. The Java interface, provides prototype methods for setting the values of the class attributes and getting these values. For example the ISO14649 entity drilling_type_operation is defined in EXPRESS as:

```
ENTITY drilling_type_operation  (* m0 *)
ABSTRACT SUPERTYPE OF (ONEOF(drilling_operation, boring_operation,
back_boring, tapping, thread_drilling))
SUBTYPE OF (milling_machining_operation);
cutting_depth: OPTIONAL length_measure;
previous_diameter: OPTIONAL length_measure;
dwell_time_bottom: OPTIONAL time_measure;
feed_on_retract: OPTIONAL positive_ratio_measure;
its_machining_strategy: OPTIONAL drilling_type_strategy;
END_ENTITY;
```

The equivalent Java interface for the above entity would be:

```
public interface drilling_type_operation_interface extends
milling_machining_operation_interface {
void set_cutting_depth(length_measure_interface parameter);
void set_previos_diameter(length_measure_interface parameter);
void set_dwell_time_bottom(time_measure_interface parameter);
void set_feed_on_retract(positive_ratio_measure_interface parameter);
void set_its_machining_strategy(drilling_type_strategy_interface
parameter);
length_measure get_cutting_depth();
length_measure get_previous_diameter();
time_measure get_dwell_time_bottom();
positive_ratio_measure get_feed_on_retract();
drilling_type_strategy get_its_machining_strategy();
}
```

The attributes that are aggregated with LIST[?:?] and SET[?:?] in EXPRESS are both translated into arrays in Java.

After defining the interface, the class can be defined. In translating EXPRESS entities to Java, two types of entities can be identified: the instantiable entities and the abstract entities. The definition of abstract entities includes "ABSTRACT SUPERTYPE OF". The definition of instantiable entities does not include the "ABSTRACT" keyword. In the object oriented paradigm, abstract classes are classes that can not be instantiated and therefore, a good representation for abstract EXPRESS entities.

The equivalent Java class for the above definition would be:

```
public abstract class drilling_type_operation extends
milling_machining_operation implements drilling_type_operation_interface {
private length_measure_interface cutting_depth=null;
private length_measure_interface previous_diameter=null;
private time_measure_interface dwell_time_bottom=null;
private positive_ratio_measure_interface feed_on_retract=null;
private drilling_type_strategy_interface its_machining_strategy=null; }
```

In addition the Java class must provide implementations for all of the methods defined in the interface. For example:

```
void set_its_machining_strategy(drilling_type_strategy_interface param){
its_machining_strategy=param; }
drilling_type_strategy_interface get_its_machining_strategy() {
return its_machining_strategy;}
```

To ensure that no method remains unimplemented, each Java class is defined so that every explicit attribute, that is every attribute that is defined in the class or one of the parents or one of the interfaces, has setter and getter methods. These are methods that set a specific attribute to a value given as a method parameter and report the values of the attributes.

Furthermore each instantiable class's constructor allows all the attributes to be set when creating a new object based on the class. For the *machining_workingstep* entity, for example, the following constructor is implemented:

```
public machining_workingstep(identifier_interface its_id,
elementary_surface_interface its_secplane, manufacturing_feature_interface
its_feature, machining_operation_interface its_operation,
in_process_geometry_interface its_effect);
```

This can simplify creating and manipulating objects based on the IP^{3}AC classes. The IP^{3}AC definition of the ENTITIES that are not subtypes of any other ENTITY, extend the abstract

class *express_entity*. Consequently input and output methods can be defined generically across the platform to allow a unified view of STEP-NC information.

Each translated EXPRESS entity has an input method to read the information through the data access interface from the data repository and assign the correct values for each attribute. The output method will provide the means for storing updated data from the attributes in the object back into the data source as seen in figure 6.5.

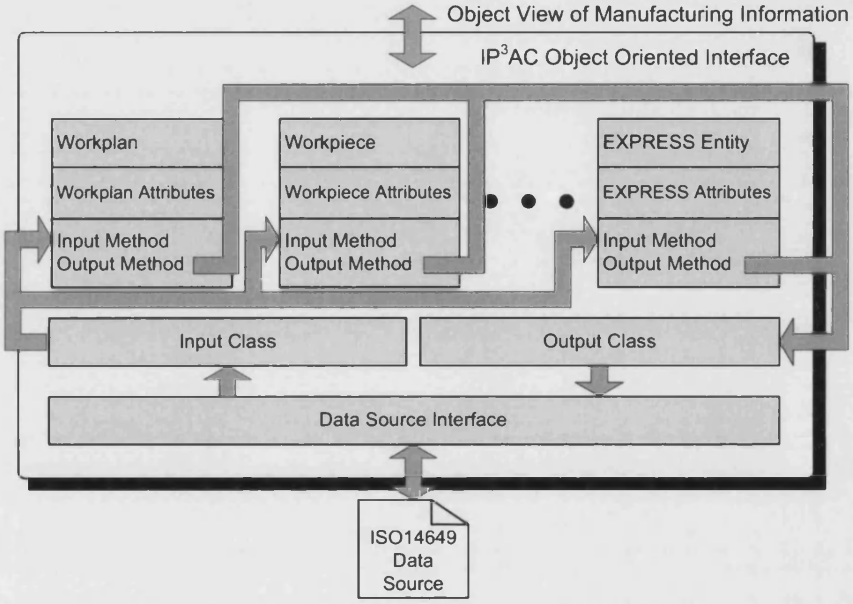


Figure 6.5 - Generic input and output methods for IP[3]AC classes

6.4.2. Types

EXPRESS allows the definition of various kinds of "TYPE"s. These include *simple types*, *enumeration types* and *select types*. These are defined as follows in IP[3]AC:

(i) Simple Types

The general definition of simple types takes the following form in EXPRESS:

```
TYPE Type_name = Type_definition;
END_TYPE;
```

Where *type_name* is the name of the type being defined and *type_definition* is another type or a primitive type. The primitive types in EXPRESS are INTEGER, STRING, BOOLEAN and REAL among others.

In IP^{3}AC the primitive types are defined as classes extending *express_type* and implement interfaces that are designed for allowing multiple inheritances when necessary.

For example the class INTEGER is defined as follows:

```
public class INTEGER extends express_type implements INTEGER_interface {
private int its_value;
public INTEGER() {}
public INTEGER(int param) {its_value=param;}
public void setValue(int param) {its_value=param;}
public int getValue() {return its_value;}
}
```

and the interface INTEGER_interface is defined as follows:

```
public interface INTEGER_interface {}
```

As the interface is defined only for multiple inheritance purposes no methods are defined within the brackets.

Non-primitive simple types extend primitive simple types or other non-primitive simple types and have no additional methods or attributes. For example *time_measure* is defined as follows:

```
public class time_measure extends REAL implements time_measure_interface{
public time_measure() {super();}
public time_measure(real param) {super (param);}
}
```

(ii) Enumeration Types

Enumeration types are utilised within EXPRESS to define elements where the value is chosen from a pre-specified set of values. For example *hand_of_tool_type* is defined as follows in ISO14649:

```
TYPE hand_of_tool_type = ENUMERATION OF (left,right,neutral);
END_TYPE;
```

Within ISO10303-21 coding, parameters of these types are shown as "*.name.*", with name being one of the values in the list. For example ".neutral." is a valid ISO10303 representation of the above type. The IP^{3}AC class representing enumeration types is defined based on this notion and values can chosen by their names or an integer index.

(iii) Select Types

Within an EXPRESS schema, these types are used to show that a parameter can be based on a number of types or entities. For example in ISO14649 in the entity *trimmed_curve* the attribute *trim_1* is a set of *trimming_select. trimming_select* is defined as follows:

```
TYPE trimming_select = SELECT (cartesian_point, parameter_value);
END_TYPE;
```

This means that *trim_1* can be a set of either *cartesian_points* or *parameter_values.* In IP^{3}AC each select type is defined as an interface, with all the classes that are in the *select type* implementing that interface. Effectively where *trimming_select* is called, any of the classes that have implemented *trimming_select* can be used. *Cartesian_point* and *parameter_value* therefore, both implement the *trimming_select_interface.*

6.4.3. Constants

Constants can be defined as those in the *express_entity* class and used throughout IP^{3}AC if necessary.

6.4.4. Functions

Functions are defined in a non object-oriented manner in EXPRESS as they are generic definitions not tied to any particular entity. To preserve the knowledge contained within these functions, it is possible to implement them as methods in the *express_entity* class.

6.4.5. Where, Derive, Inverse

In each ENTITY in addition to attributes and inheritance links, a number of other definitions can be added in EXPRESS. These include WHERE, DERIVE and INVERSE. These elements try to capture meta-data and knowledge within the framework to ensure that a population

created within the domain is valid. These rules can be implemented as validation methods in each IP^{3}AC class to ensure the integrity of the information.

6.4.6. Optional

In an EXPRESS schema, attributes can be defined as optional or obligatory. It is possible to capture this knowledge in IP^{3}AC in the form of constructors that do not allow an object to be created unless all the non-optional attributes are set. Alternatively the requirements can be relaxed and included as a part of the validation methods inside the classes.

6.5. EXPRESS Translator

Instead of manually translating each EXPRESS definition into a Java class definition in IP^{3}AC, an automatic code generator has been utilised to create IP^{3}AC. The functional representation of the code generator named the EXPRESS translator is shown in figure 6.6.

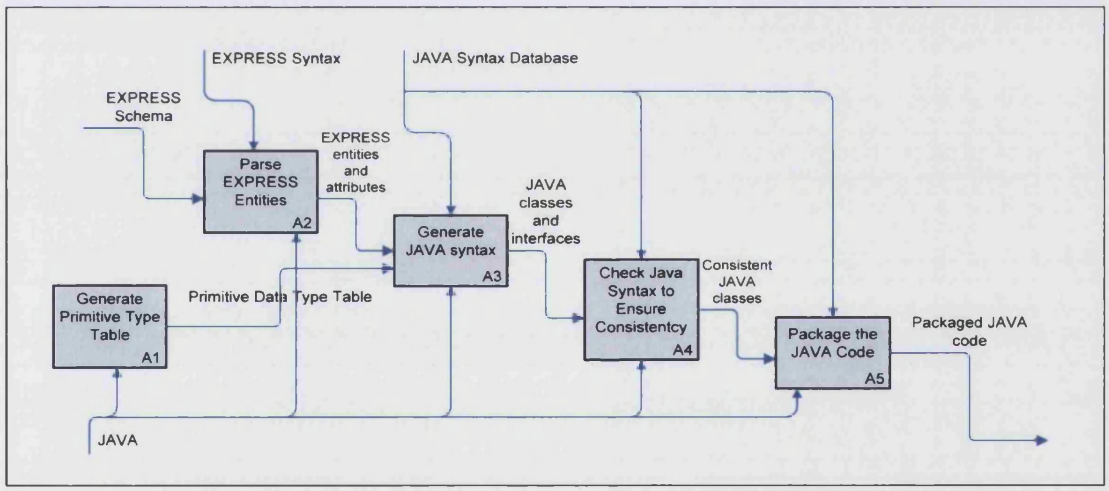


Figure 6.6 - IDEF0 representation of the EXPRESS Translator

The code generator starts by defining a table of primitive data types in EXPRESS and creates their corresponding Java classes. The EXPRESS schema (i.e. the schema for ISO14649) is parsed with the entities, attributes and various types identified. These elements are then passed on to a Java syntax generator that creates the corresponding Java representations for each EXPRESS element. The generated Java class and interfaces are then checked and validated to ensure the generated platform is consistent in its definitions. Finally the classes and interfaces

that have been confirmed as being consistent are packaged in a Java code package. Figure 6.7 shows a detailed view of the Java syntax generation activity.

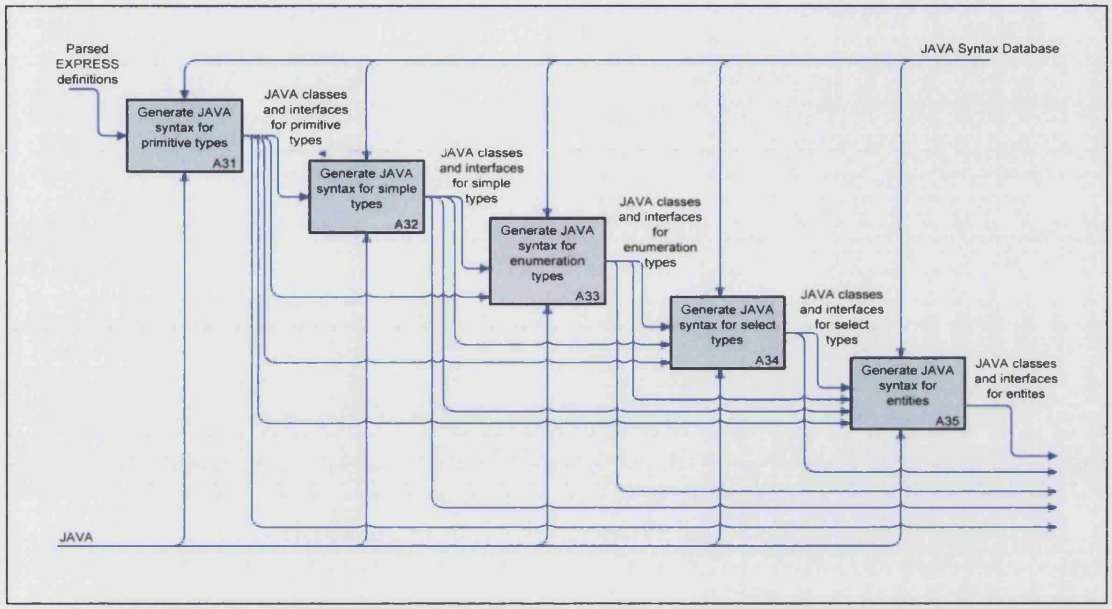


Figure 6.7 - IDEF0 representation of activity A3 - Java syntax generation

In this activity Java representations for EXPRESS elements are generated according to the specifications in section 6.4. In the first activity, interfaces and classes for the primitive types are created. These are then extended to create the Java syntax for simple types. Enumeration and select types are then generated, implementing the interfaces defined as a result of the processing of primitive and simple types. Finally the Java syntax for the entities is generated implementing all interfaces for the previous elements. The output of the activity is a collection of Java classes and interfaces.

6.6. Advantages and Disadvantages of IP^{3}AC

IP^{3}AC provides an early binding of STEP-NC entities to Java classes. These bindings are automatically generated. This approach creates a number of significant advantages and a number of disadvantages over the alternative technique of using a SDAI implementation.

6.6.1. Simple Manipulation of Manufacturing Information in the Java Environment

Using IP^{3}AC the generated classes and interfaces are directly accessible in the object space provided by the Java environment. The objects generated based on the classes can be organised in collections, serialised and manipulated with ease in the Java runtime environment. Figure 6.8 shows an example where a Java "ArrayList", essentially an ordered collection of objects, is utilised to generate an ISO10303-21 representation of the population.

While SDAI implementations offer ISO10303-21 generation abilities, these are not open and modifiable by the developer. Any optimisations will therefore need to be implemented by the SDAI implementation vendor and the developer cannot control the process.

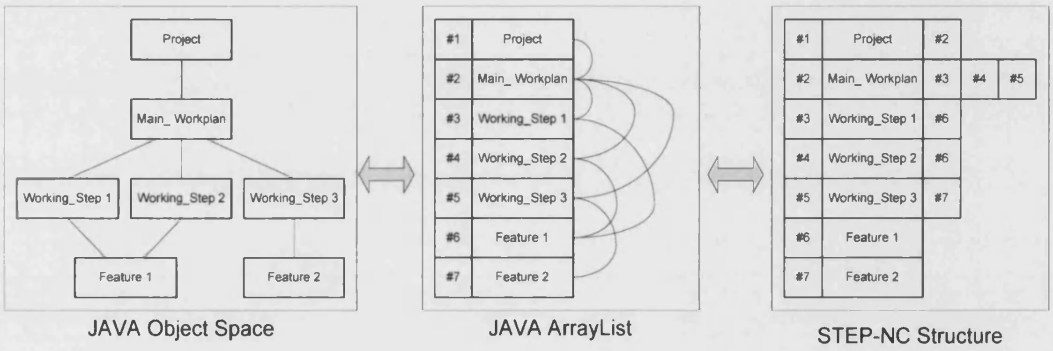


Figure 6.8 - Storage and retrieval of the STEP-NC Program Structure

6.6.2. The Ability to Define Additional Methods during Automatic Code Generation

As IP^{3}AC is automatically generated, it is possible to define additional methods while generating the platform. For example it is possible to define aggregate reporting methods for all classes. Such a method can report all of the attributes of a specific class with their respective types as a collection of strings. This method can be useful in developing applications like tree viewers that display a hierarchical view of manufacturing information. Figure 6.9 shows a tree viewer application developed using IP^{3}AC. A recursive algorithm is used to generate the tree, where first the root of the tree is created using the project object instance. For each class the aggregate attribute reporting method is invoked and all attributes are enumerated. Should the attribute be a primitive, then a node showing the value is created. If the attribute is an EXPRESS entity, in addition to the creation of the node representing the entity, the process of going through the attributes is repeated for that entity. When the

execution of the algorithm is finished, the tree structure is complete and can be displayed. Through the use of aggregate attribute reporting methods defined in IP^{3}AC, the tree viewer application has been developed efficiently by using only 26 lines of Java code. Figure 6.9 shows a screenshot of the application displaying the tree view of an ISO14649 part programme.

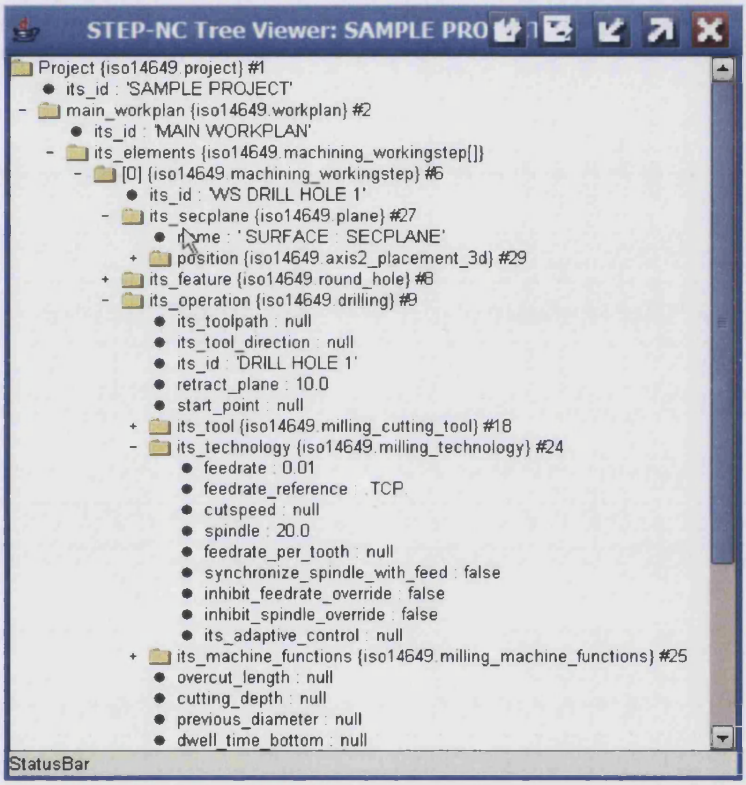


Figure 6.9 - IP^{3}AC STEP-NC Tree Viewer

6.6.3. The Ability to Utilise Existing Libraries to Manipulate Manufacturing Data

In addition to creating extra methods during the automatic code generation, it is possible to implement extra interfaces for the classes to link to the existing Java libraries. To further test this ability, a 3D part viewer was developed, relying on Java3D libraries. Figure 6.10 shows the 3D viewer displaying a sample part.

Each class that extends the feature class was enhanced to implement a solid generation interface that enables the feature class to report its 3D representation in the form of a set of

triangles. The workpiece class was augmented to implement the same interface. The 3D representations of the features were then subtracted from the 3D representation of the workpiece using a Boolean 3D library. The resulting complex shape was then added to the Java3D 3D object tree and displayed.

The user can manipulate the viewing angle by controlling the mouse as defined in the Java3D libraries without the need of extensive programming. Furthermore colour changes, zooms and pans and lighting settings are all controllable through the predefined methods in Java3D and can be accessed with no requirement for manufacturing specific programming.

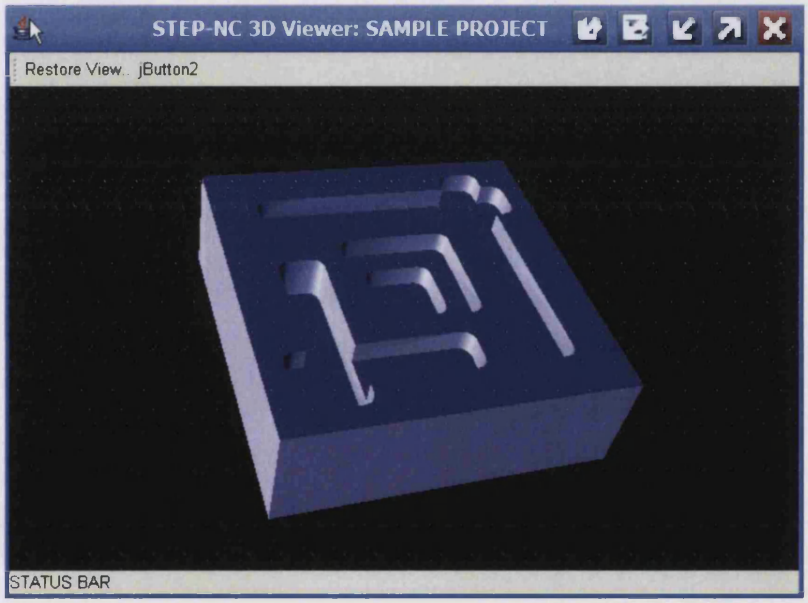


Figure 6.10 - STEP-NC 3D part viewer

6.6.4. Extensibility to Provide a Foundation for Manufacturing Software

One of the IP^{3}AC's design disadvantages is that, should the EXPRESS definitions within the manufacturing Interlingua change, the platform needs to be regenerated to reflect the changes. SDAI allows dynamic updates to be made to the EXPRESS schema. The author believes that this disadvantage is not significant as changes in the standards happen very infrequently.

Furthermore the design of the EXPRESS translator with the extensive implementation of interfaces and input/output methods, protects the underlying structure of the classes. This

means that should the standard change, the additional methods and interfaces defined by the developer will continue to function without requiring any changes. Consequently as long as the general semantics of the standard remain unchanged, user programs will not to be modified with revisions to the standard. The old version of IP^{3}AC can be replaced by the new version seamlessly.

This is evident when during the course of the research ISO14649 was amended to include elements representing inspection processes. The new version of the standard was utilised to generate a new version of IP^{3}AC by the EXPRESS translator. The modification was completed in a matter of hours after the ISO14649-16 schema had become available.

6.7. Summary

In this chapter a computational platform has been specified and designed to enable encoding of manufacturing data using the data models contained within a comprehensive manufacturing lexicon. The platform entitled IP^{3}AC utilises the data models provided by ISO14649 to store prismatic part manufacturing data. It also allows manufacturing information to be manipulated using native structures in the Java programming language.

7. Information Exchange Using Mobile Agents in the CAx Chain

7.1. Introduction

An important issue in creating an interoperable framework is to ensure the integrity of information during transfers from one resource to another. Considering that the modern manufacturing enterprise might be located in a number of geographically distant sites, maintaining a record of the semantics pertaining to the data together with the data itself becomes crucial. The current method of using data files falls short of meeting these requirements. Mobile agents provide a robust and reliable method to transfer information between CAx resources while ensuring that the semantics of the information remain unmodified during the transfer. They can also enable automatic transfer of information upon request by other resources. In this chapter a suitable agent platform for transferring manufacturing information between various CAx resources is specified, designed and realised.

7.2. Agents as Carriers of Information

As shown in chapter 5, the flow of information between CAx resources is an important consideration in designing an interoperable framework. Figure 7.1 compares various approaches in transferring information between CAx resources in a CAD/CAM/CNC chain.

File transfers are the easiest to implement. Whenever a resource is requested to provide information a file in a proprietary format is generated. These files can then be transferred using physical storage devices (i.e. floppy disks, USB memory sticks) or via a network. The problem is that when a large number of similar but slightly different resources exist on a network it is difficult to ascertain the actual semantics contained within a file.

For example two CNC controllers might save their files as ASCII text files with the extension of ".NC". These controllers might have very similar syntaxes with a number of crucial differences (i.e. G87 might be defined as a rectangular pocket machining cycle and a tap drilling cycle in another). If the files are not documented properly the semantics of the file are not easily accessible and expert operator knowledge will be required.

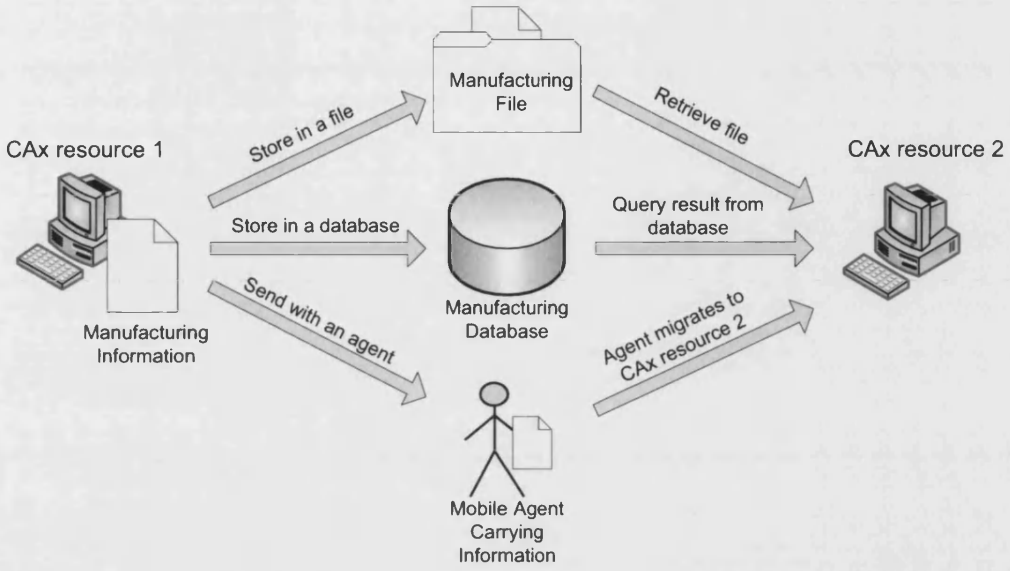


Figure 7.1 - Various approaches for information transfer from one CAx resource to another CAx resource

Using a product lifecycle management (PLM) or product data management (PDM) system to catalogue the various files on the network could alleviate this problem.

Agents are not immediately associated with information carriage and in the domain of CAx interoperability they have been mostly used for distributed process planning (see 3.4.2). An intelligent agent is defined "a computer system that is capable of flexible autonomous action in order to meet its design objectives" (Jennings and Wooldridge, 1998). With this definition it is evident that if an intelligent agent is created to maintain the integrity of a piece of information, it can take autonomous action to prevent the information from being modified to incoherence. A mobile agent consequently can be utilised to transfer information while maintaining the semantics intact.

Alternatively, a database can be used to maintain the integrity of the transferred data between resources. This however, will require all CAx resources to be able to store their information in a database with homogenised semantics. At the time of writing of this research, such a database has not materialised in the domain of CAx manufacturing although researchers have proposed such databases (Zimmerman et al. 2002).

To define a mobile agent framework, it is either designed from ground up in multi-threaded languages like Java and C++ or alternatively a ready-made agent framework is utilised and extended for a specific purpose. There are a number of Java based agent environments currently available (see Shen et al. 2006). For this research IBM's Aglets platform has been chosen as it is open sourced and one of the best known environments with support from IBM (http://www.trl.ibm.com/aglets/). Figure 7.2 shows an overview of the Aglets environment. In the Aglets environment, the abstract class "Aglet" provides the prototype for mobile agents. According to the classification in table 3.9, a basic aglet is a reactive agent and with the addition of mobility becomes a mobile agent. Each computer runs the *Tahiti Aglet Server* using the *Java Virtual Machine. Agletcontexts* which are homogenous environments for a collection of related agents, host the aglets. The *Tahiti Server* handles migration of agents automatically and provides security and ensures integrity.

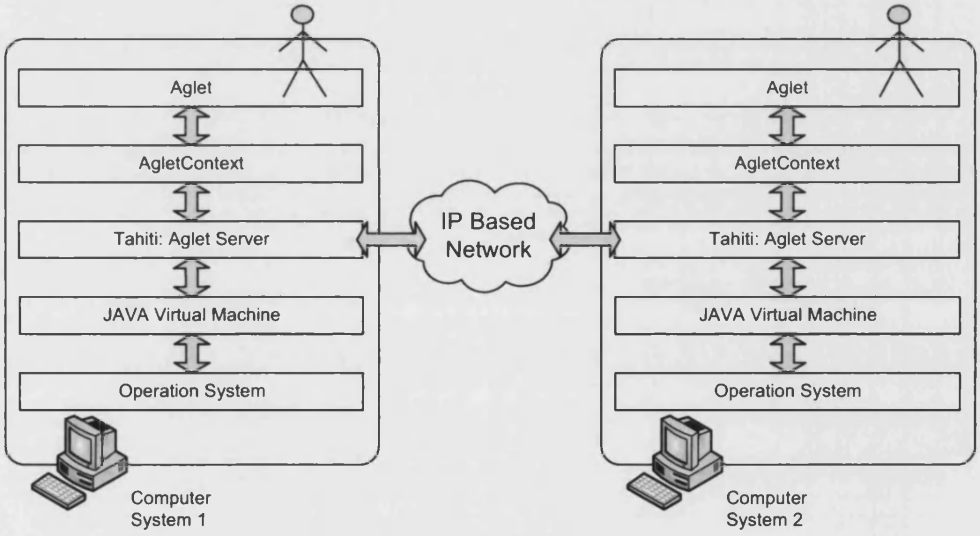


Figure 7.2 - An overview of the Aglets mobile agent environment

7.3. Information Storage in Agents

As defined in section 5.4, within the interoperable framework, any object representing proprietary formatted data should implement the *ProprietaryData* interface as seen in figure 7.3. This means that the object should be able to read manufacturing data from a data source and store it, and also be capable of writing the information to the data source. After reading

the manufacturing information from the data source, this object contains a copy of the manufacturing information that can be utilised to transfer information inside an agent. The agent responsible for carrying this information is called the carrier agent.

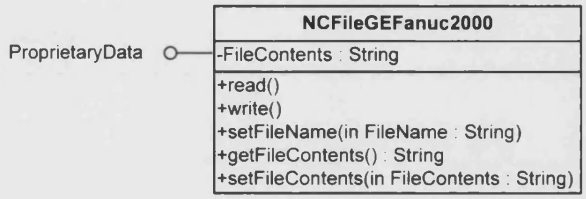


Figure 7.3 - UML class diagram of a G&M code text file data source

To ensure that the data in the object is persistent (it is storable), the interface *ProprietaryData* extends the interface *Serializable*. The *Serializable* interface, when implemented by a class, signifies that the object can have persistence and be written and read using an *ObjectStream*. An aglet implements *Serializable* and therefore any agent defined in the aglets environment can be stored to disk and loaded (as all agents extend the aglet class). Hence it is possible to define an attribute in the agent to represent the data it is charged with transferring. The UML class diagram for the carrier agent can be seen in figure 7.4.

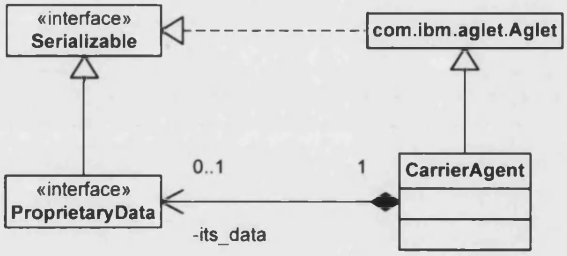


Figure 7.4 - UML class diagram of the carrier agent

7.4. Agent Mobility, Persistence and Security

By definition a mobile software agent should be able to halt execution on one computer system, transfer its state to another computer system and continue code execution where it was stopped on the first computer with the same state. The Aglets environment allows agents to despatch themselves from one computer running the aglets server to another using its unified

resource locator (URL) address. It is also possible to define complex itineraries for agents that need to visit a number of computer systems in a specific order to achieve their goals.

Any agent that is developed based on the *aglet* class despatches itself to another computer when its *dispatch* method is called. In the interoperable framework where CAx resources are interconnected using a network, the data carrier agent is required to have the capability to find the URL of the destination CAx resource address that requires the information it is carrying.

In order to achieve this, in addition to the carrier agent, two more types of agents have been defined within the framework, namely the directory agent and the monitor agent. The monitor agent observes a CAx resource and interprets requests for information and also makes queries of the resource for data as seen in figure 7.5.

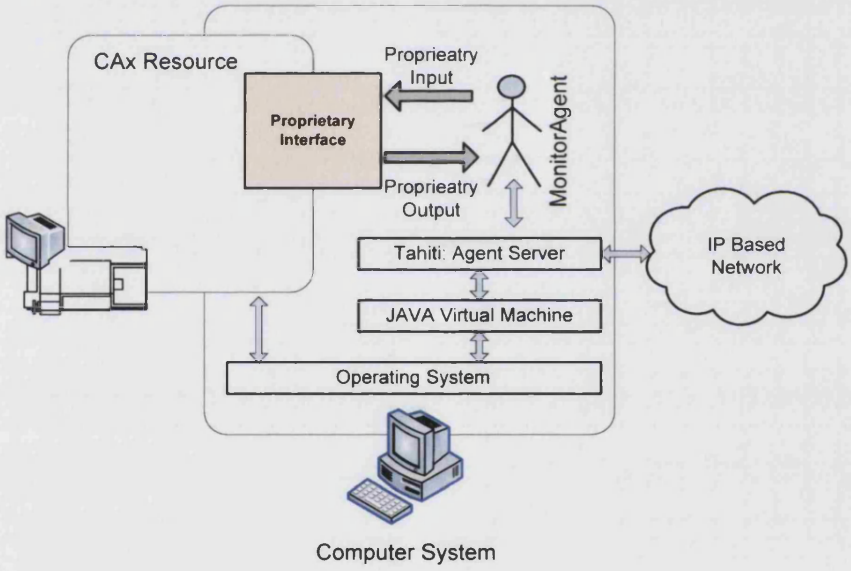


Figure 7.5 - Overview of the monitor agent.

The directory agent maintains a list of all of the CAx resources that are available in the enterprise together with their URLs. To achieve a high degree of security, the directory agent is hosted on a computer system at the heart of the manufacturing enterprise where information transfers are regulated. Figure 7.6 shows an example of the directory agent listing the CAx resources in a global manufacturing enterprise.

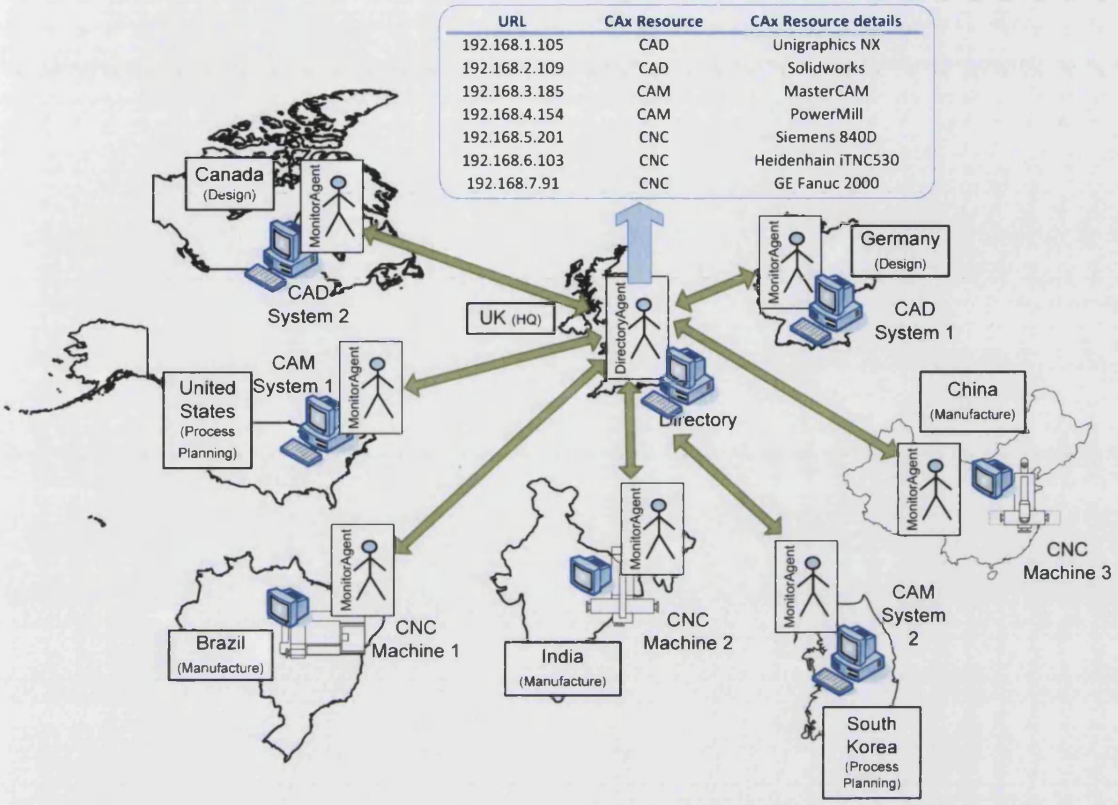


Figure 7.6 - An example of agent interactions in a global manufacturing enterprise.

The aglets environment addresses the low-level security issues by running the agents in the protected *agletcontext* environment and accessing them by using proxies as seen in figure 7.7. Foreign programs cannot exchange information with this environment without having the necessary security certificates. This protects the agent framework from possible intrusions by potentially dangerous software.

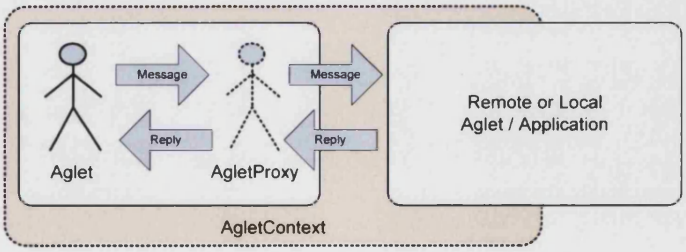


Figure 7.7 - Aglet proxy used to send and receive messages to aglets

The persistence of agents is also addressed by the aglets environment. It automatically maintains a copy of the agents in transfer in case a communication link is severed and retries transmission when the link is established again.

Carrier agents, together with monitor agents and the directory agent, allow the transfer of proprietary formatted information from one CAx resource to another within the interoperable framework. It is however necessary to implement an agent based representation for the abstraction work presented in chapter 5. The abstraction agent is defined to address this issue. An abstraction agent carries the necessary logic to translate data from the standardised format to the proprietary format and vice versa. This is illustrated in figure 7.8.

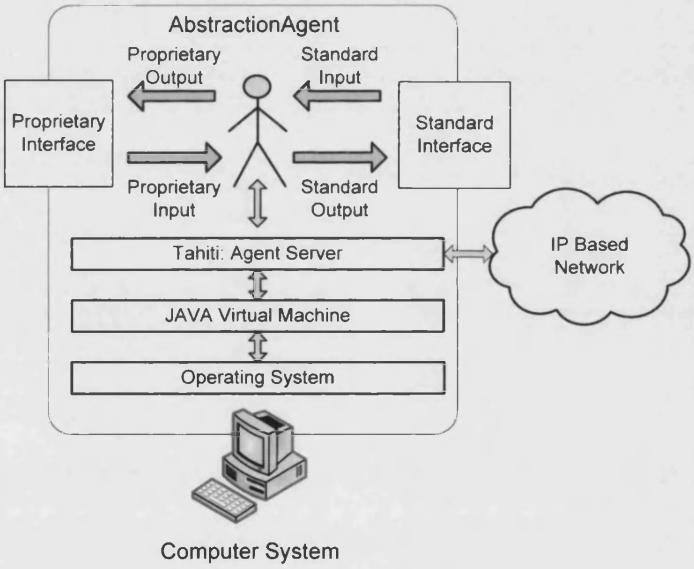


Figure 7.8 - Overall view of the abstraction agent

7.5. A Mobile Agent Based Information Transfer Framework

The information transfer framework is created on the basis of an IP based network (Clark, 2003) connecting all CAx resources together. This could be a local area network or a globally distributed network on the internet. The network, henceforward called the "manufacturing network", allows the connected devices to exchange messages based on the various protocols provided by the internet protocol.

The complete agent based information transfer framework is created using the four agents (i.e. carrier agent, abstraction agent, directory agent and monitor agent) described in 7.3 and 7.4. In the first initialisation of the directory system, the directory agent is created and awaits monitor agents to register themselves.

Upon connection of a CAx resource to the manufacturing network, the computer system hosting the resource creates a monitor agent, this agent first registers itself with the directory agent so that the network is aware of the new resource and knows where this resource is located (using the URL). The monitor agent then creates an abstraction agent for the resource that it is monitoring and sends it to the directory. With each change in the resource configuration, this agent is recreated and resent to the directory so that the directory always has a current version of the abstraction logic for that particular resource.

The sequence is illustrated in figure 7.9 using a UML sequence diagram.

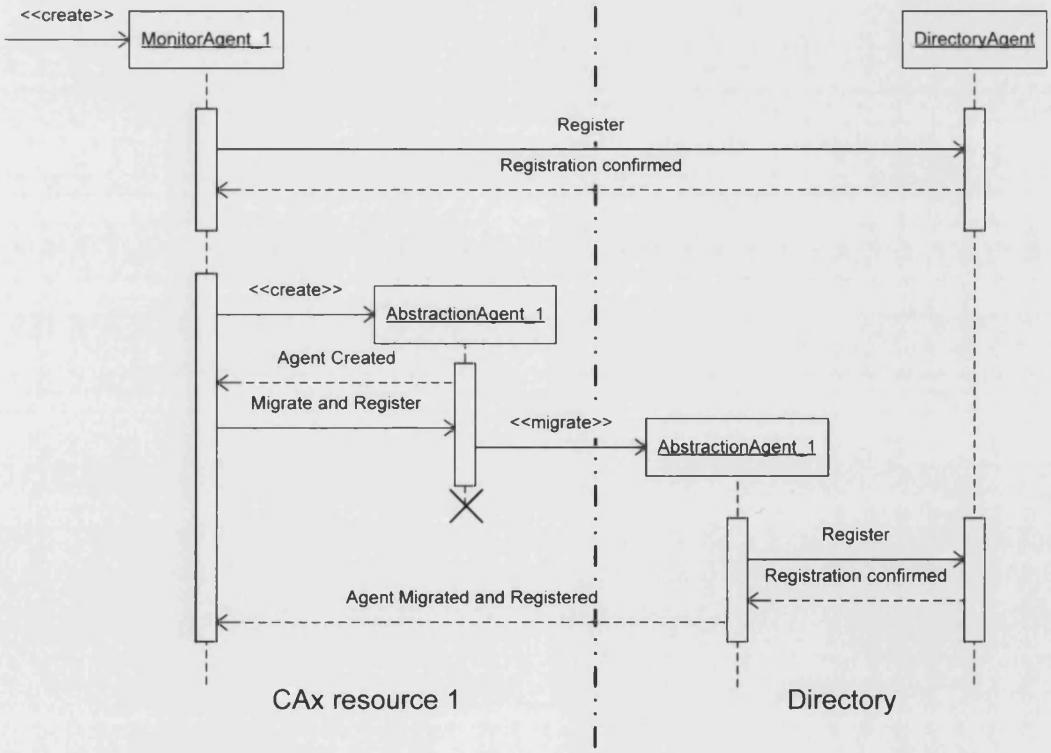


Figure 7.9 - UML sequence diagram showing the initial agent registration sequence

After all CAx resources are registered in the directory, information transfer can begin. The first phase of the transfer of information from CAx resource 1 to CAx resource 2 is to transfer the abstraction logic to resource 1. In phase 2 the information which has been converted to the format understandable for CAx resource 2 is transferred to the resource using a carrier agent. Figure 7.10 shows the phase 1 of information transfer, where the abstraction logic for CAx resource 2 in the form of *AbstractionAgent_2* is requested from the directory agent, who instructs *AbstractionAgent_2* (Which was created when CAx resource 2 registered itself with the directory agent) to clone itself and send the copy to CAx resource 1 at the URL that is in the resource table kept by the directory agent.

Upon arriving at the destination the agent will report its arrival to the monitor agent for resource 1, which then signals the beginning of the second phase of information transfer.

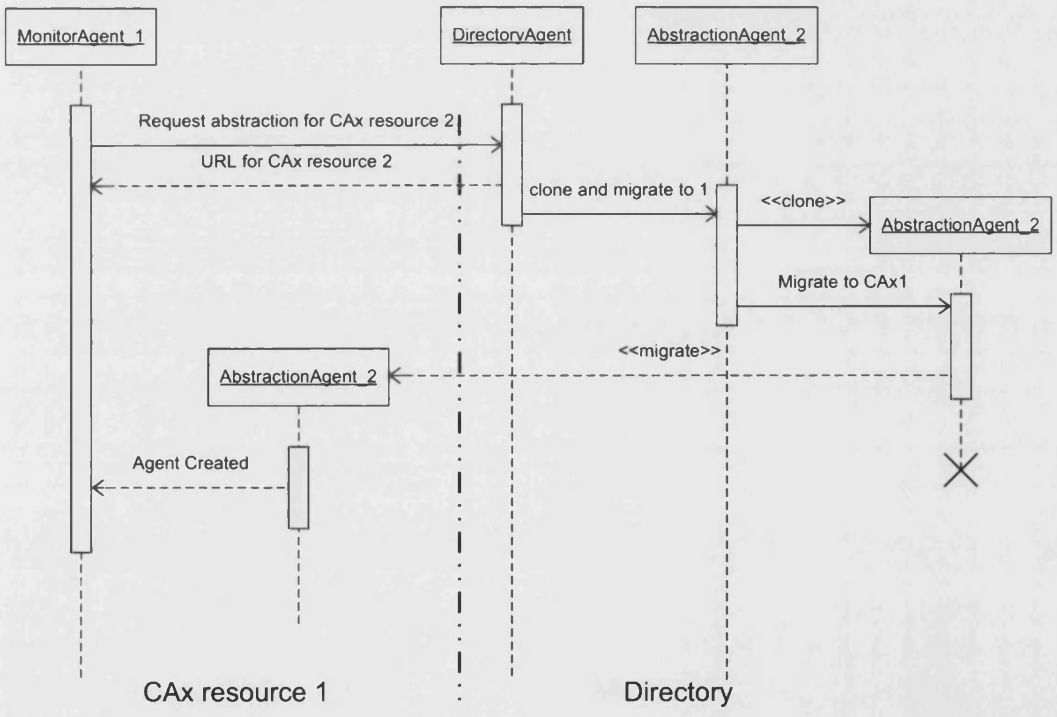


Figure 7.10 - UML sequence diagram showing information transfer phase 1 - abstraction agent migration

The sequence for phase 2 of the information transfer from CAx resource 1 to CAx resource 2 is shown in the UML sequence diagram in figure 7.11. The monitor agent in resource 1 first creates a new abstraction agent for resource 1 to reflect the latest changes in the abstraction logic. The proprietary item of information that needs to be translated is then passed on to the abstraction agent and the standardised data generated by the agent is received back by the monitor agent. The abstraction agent for resource 1 is then terminated.

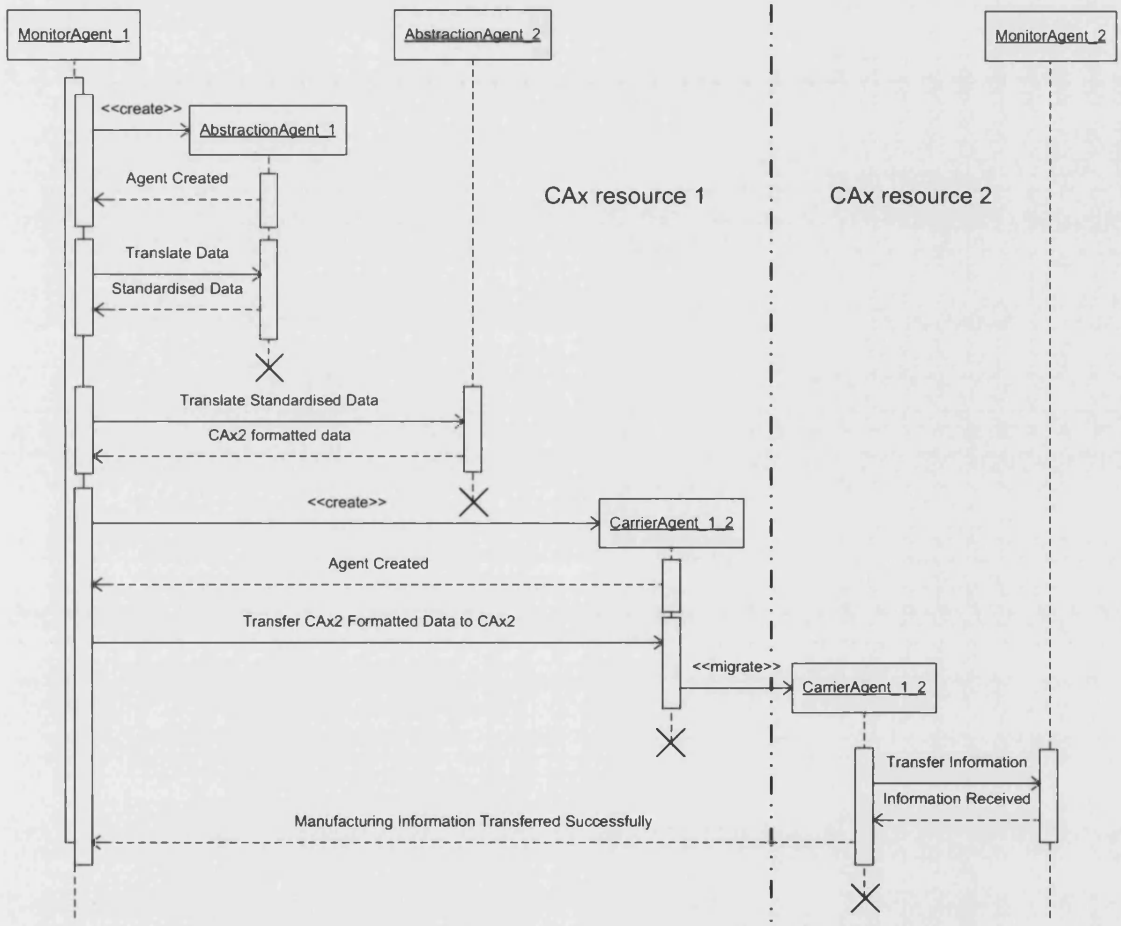


Figure 7.11 - UML sequence diagram showing information transfer phase 2 - carrier agent migration

Standardised information is then passed on to the abstraction agent for resource 2 who migrated to resource 1 in phase 1 of the data transfer. That agent generates data suitable for CAx resource 2 based on the standardised data. This proprietary data is returned to the monitor agent and abstraction agent is terminated.

The monitor agent then creates a carrier agent and sends the proprietary data formatted according to the requirements of CAx resource 2 to it. It then instructs the carrier agent to migrate to CAx resource 2 (whose URL was received from the directory in phase 1) and passes the information to the monitor agent.

The carrier agent migrates and transfers the information to the monitor agent for resource 2. Before termination, the carrier agent sends a message to the monitor agent on resource 1, informing it of the successful completion of transfer.

Based on these sequences and the design of the agents, the mobile agent based information transfer communication element of the interoperable framework is constructed. The overall view of the framework can be seen in figure 7.12. In this figure different implementations for placement of the agent server is presented. When the resource has enough computational power to host the agent systems, the monitor agent is run on the resource. If the resource does not have enough processing capacity (i.e. legacy CNC controllers) the agent server is hosted on an interface PC.

The final specification of the agents as defined in the framework can be seen in the form of a UML class diagram in figure 7.13. Please note that only the methods that override the method definitions in the Aglet base classes are identified on the UML class diagram.

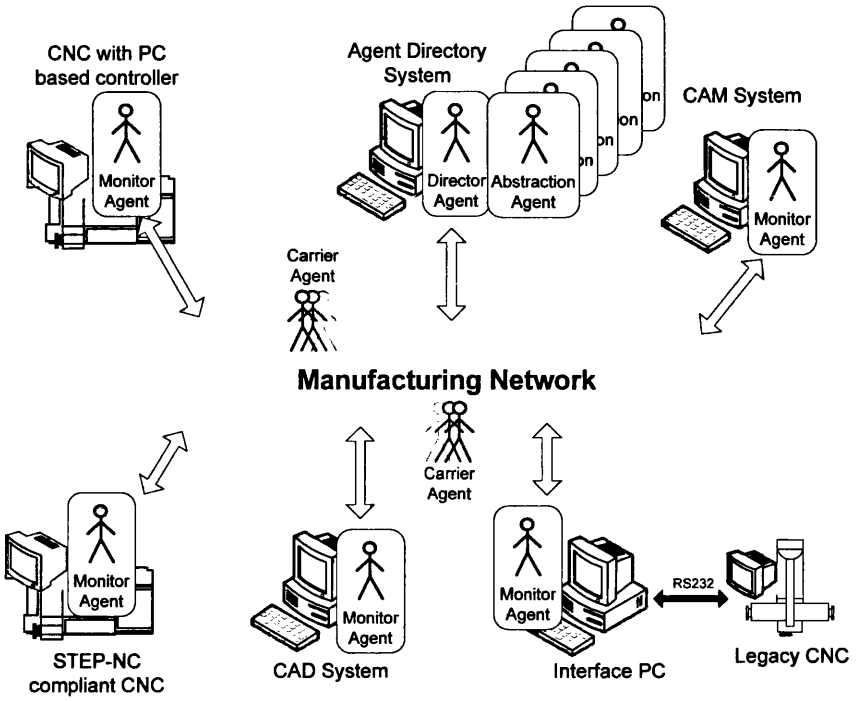


Figure 7.12 - Overall view of the mobile agent based information transfer framework

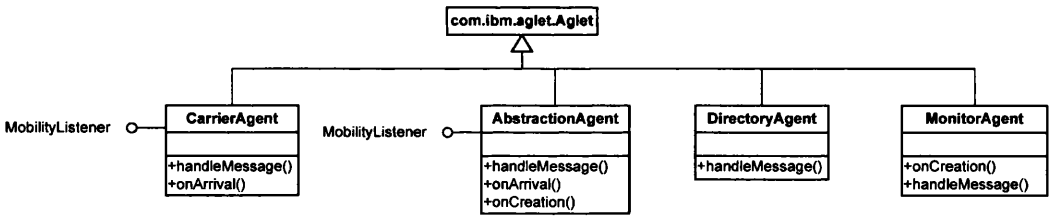


Figure 7.13 - UML Class diagram of the agent based information transfer framework

As seen in sequences in figures 7.9, 7.10 and 7.11 various messages should be handled by each agent with various functionalities associated with each message. Table 7.1 lists the important messages, the responsible agents, message parameters and a description of the functionality of the agents when the message is received, in the framework.

7.6. Advantages of Using Agents for Information Transfer

While development of mobile agents for transferring manufacturing information might seem like a complicated solution, there are several advantages that can be gained by employing the agents over the current method of exchange information in data files.

Table 7.1 - Fundamental agent messages in the mobile agent based information transfer framework

Message	Responsible Agent	Parameters	Description
Register	DirectoryAgent	Registering Agent, Directory URL	Registers an agent with the directory agent
Request Abstraction Agent	DirectoryAgent	CAx resource for which abstraction is required	Sends the URL and monitor agent address of the destination resource as reply and instructs the appropriate abstraction agent to clone and migrate to the initiating resource
Migrate and Register	AbstractionAgent	URL, Registrar agent	The abstraction agent migrates to the context specified in the directory URL and registers itself with the registrar agent
Translate to Standard	AbstractionAgent	Proprietary Data	Replies with the standardised information based on the semantics contained in the input parameter and the CAx resource model
Translate to Proprietary	AbstractionAgent	Standard data	Similar to above, replies with proprietary information instead
Transfer Information	CarrierAgent	Proprietary data, URL, Receiving agent	Instructs the carrier agent to take a copy of manufacturing data, migrate to the context at the destination URL and pass the data to the receiving agent
Send information	MonitorAgent	Requester, Information query, Reference	The monitor manipulates the other agents to generate a suitable reply to the query and send it to the requester
Receive information	MonitorAgent	Sender, Information, Reference	The monitor receives information together with the identity of the sender of the information

7.6.1. Monitoring

Due to their distributed execution, mobile agents are excellent for monitoring the status of objects on a network of computers. In a global manufacturing enterprise where a plethora of machine tools, cutting tools, CAx systems, fixtures and other entities need to be tracked across multiple geographical locations, agents provide the necessary abilities to monitor the resources. A monitor agent for example, can keep track of a specific machine tool, always knowing what configuration of cutting tools is at use on the machine as well as tools that are available to the machine for changing.

The monitor agent coupled with resource abstraction provides a standardised agent based representation of resources on the CAx network. As a result the entire enterprise's condition can be ascertained by making a simple query of all the monitor agents.

7.6.2. Data Collection from a Variety of Data Sources

Mobile agents can go through a pre-specified itinerary to a number of computer systems, make enquiries of their databases and combine the information to achieve their goals. An agent looking for a specific cutting tool for example, can travel from CNC to CNC to find the tool. Having found the tool, it can query the automatic tool transfer systems nearby to ascertain whether they could provide the necessary capability to take the tool to the agent's originator. It can then instruct the automatic tool loaders and return to its home to report success, tool location and estimated arrival time.

7.6.3. Contingency In Case Of Failed Transfers and Semantic Integrity

The agent based information transfer framework, allows different types of CAx resources that use various types of storage to write their data to communicate with each other with no concern about the underlying technology. The mobile agents treat the information gathered from file sources in the exact same manner that they handle information retrieved from a database. This provides a high degree of abstraction of computer hardware resources that leads to semantic integrity of information being maintained on the software level. Failed transfers are also well catered for; an agent is always expected to report its arrival at its destination, something that is difficult to achieve with the other techniques. In case of not receiving the confirmation of safe arrival, the originating system can simply send another copy of the information.

7.6.4. Decrease in Bandwidth Requirements

Using agents can help minimise the bandwidth requirements in the interoperable CAx network. Considering that the resource abstractor requires up-to-date knowledge of the resource at anytime, it is essential for it to be located as close to the resource as possible as it needs to be updated with the status of the CAx resource regularly.

Therefore if another resource wants to send information, the abstractor will need to work remotely. Having in mind that many abstractors will require human input as part of their process, this will generate a considerable amount of network traffic.

In the agent based framework, when data transfer is initiated the latest version of the abstraction logic is packaged in an agent and sent to the originator of information. Further interactions are handled locally on the originating CAx resource and does not generate network load.

7.7. Summary

In this chapter the third and the final element of the Interoperable CAx framework, namely the communication mechanism has been specified and designed. This is in the form of a mobile agent based communication system that provides security and reliability for information transfers. The agents' ability to maintain state as well as data ensures the integrity of the semantics of the transferred information. This method is more reliable, more robust and more secure than the current approach of transferring data files.

8. Realisation of Interoperable CAx

8.1. Introduction

A prototype interoperable CAx system has been implemented based on the framework presented in the previous four chapters. This chapter outlines the realisation process and describes the various functionalities of the prototype in presenting the advantages of the interoperable CAx framework. Various design considerations and limitations are also specified.

The realisation of the prototype interoperable system is documented using three CAx resources: two modern commercial CNC controllers together with a prismatic STEP-NC compliant CAD/CAM system. The commercial CNC controllers represent the indirect abstraction approach while the developed CAD/CAM system shows the potential of the framework for direct abstraction of future CAx resources. Figure 8.1 shows the components of the framework that have been selected for implementation in the prototype.

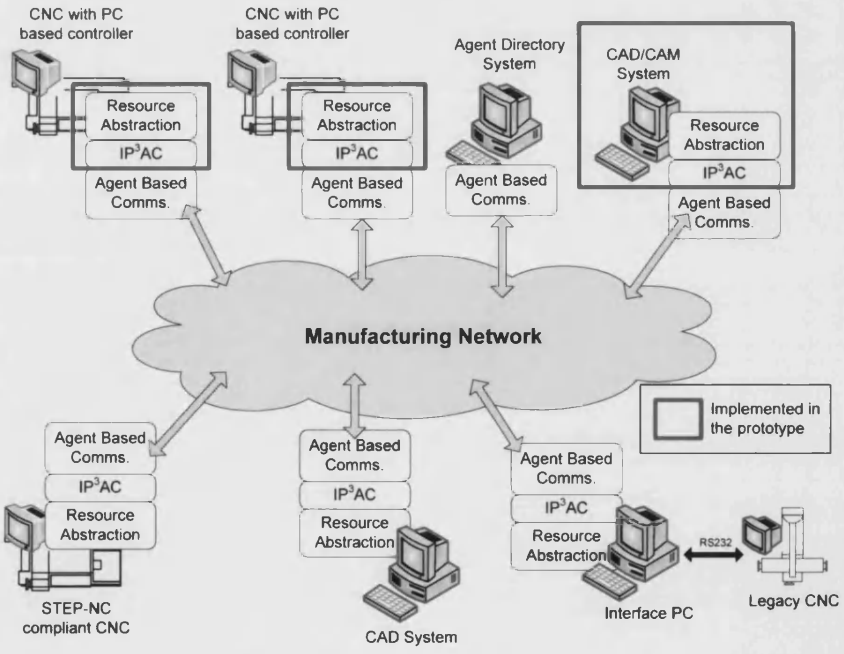


Figure 8.1 - Interoperable CAx framework components implemented in the prototype

8.2. Development of the Data Structures Using IP^{3}AC

The first step in the prototype development for the interoperable CAx framework has been the generation of IP^{3}AC classes based on the standard manufacturing language. As mentioned in section 6, ISO14649 provides the necessary information constructs to represent the manufacturing information for prismatic components. In addition to the classes representing the entities in ISO14649, a number of auxiliary classes have been generated as well to make building programs on the IP^{3}AC foundation simpler. These classes include *express_entity, input* and *output.* The *express_entity* class is the basis for all IP^{3}AC generated classes and provides the blueprint for the necessary functionality for each class.

Figure 8.2 shows the express_entity class as a UML class diagram. Here four methods have been defined, each of which, must be implemented by the classes extending the express_entity. These methods are specified in table 8.1 where the purpose of each method is described.

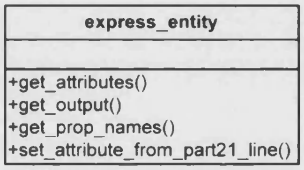


Figure 8.2 - UML Class diagram for the *express_entity*

Table 8.1- Methods defined in the *express_entity* class

Method Name	Function
get_attributes	Returns all of the attributes of any express entity in an ordered list. Each attribute is returned in its original container and therefore any manipulations in the list will reflect changes in the reporting object.
get_output	Returns the ISO10303-21 representation of the express entity. As a parameter, a list of all the entities in the population is required to construct the hash numbers.
get_prop_names	Returns an ordered list of standard attribute names. These can be used to access properties from the get_attributes by attribute name instead of index.
set_attribute_from_part21_line	This method sets the object state to reflect a line of code in an ISO10303-21 encoded STEP-NC file

The *input* class provides the necessary functionality to read STEP-NC ISO10303-21 encoded files. The UML Class diagram can be seen in figure 8.3.

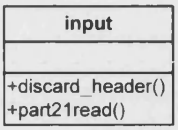


Figure 8.3 - The IP[3]AC input class

This class provides two methods: a *discard_header* method which separates the computer understandable "DATA" section of the ISO10303-21 encoded file from its "HEADER" section and a *part21read* method. The *part21read* method utilises the *set_attribute_from_part_21_line* method in the entities to read a text encoded STEP-NC file and creates the corresponding objects.

Finally the output class provides the necessary methods to create a STEP part 21 file from the data in the Java object space. The UML class diagram for this class can be seen in figure 8.4.

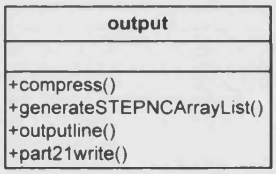


Figure 8.4 - UML class diagram for the output class

The methods that are provided by this class can be seen in table 8.2.

Table 8.2 - Methods provided by the output class

Method Name	Function
compress	This method compresses the STEP-NC data so that no empty lines exist in the text encoded output. This is useful after edits where entities have been deleted.
generateSTEPNCArrayList	Starts with a *project* entity and drills down the hierarchy, adding all entities into an ArrayList.
outputline	Creates one ISO10303-21 line of output based on one entity.
part21write()	Uses the above method to create the complete ISO10303-21 representation of the STEP data.

These classes together with EXPRESS based classes provide the information encoding that is required by the interoperable framework.

8.3. Development of the CAx Resource Abstractors

In the prototype, three CAx resource abstractors have been implemented. Two CNC abstractors for the Siemens 840D CNC with ShopMill and the Heidenhain iTNC530 CNC have been developed. In addition an abstractor for a fully STEP-NC compliant CAD/CAM system developed from scratch during the research is utilised together with the CNCs to demonstrate both indirect and direct abstraction of CAx resources.

8.3.1. The Interoperable CAD/CAM System

The interoperable CAD/CAM system has been developed in Java using the IP^{3}AC foundation and incorporates both the treeviewer and 3D viewer applications from section 6. The program which adds a 2D viewer to simplify prismatic process planning allows the user to create STEP-NC process plans and generates the ISO10303-21 encoded files for the part. Figure 8.5 shows the user interface of the interoperable CAD/CAM system.

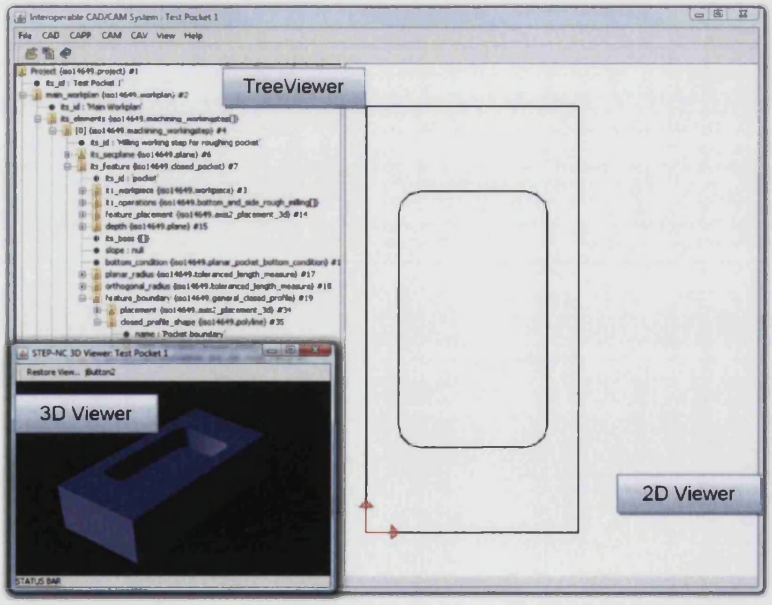


Figure 8.5 - The interoperable CAD/CAM system user interface

The system has been designed with the capability to handle a number of STEP-NC features namely planar facing, closed pockets and round holes. Each of these features can be created by choosing the appropriate options from the menus. As an example, figure 8.6 shows the hole creation interface.

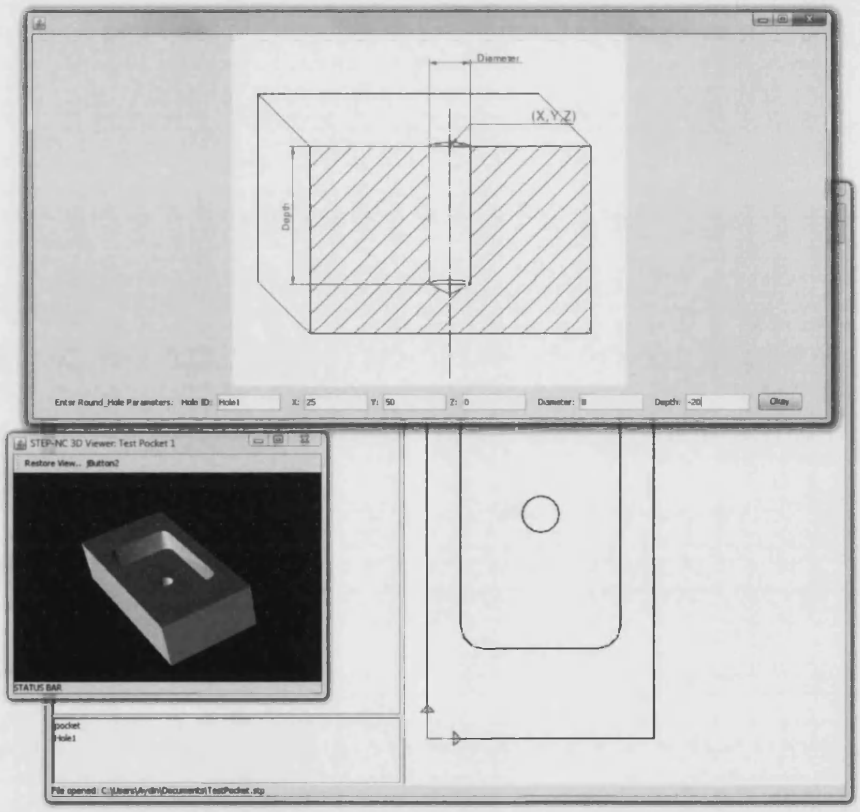


Figure 8.6 - The hole creation interface in the interoperable CAD/CAM system

The interoperable CAD/CAM system maintains manufacturing data using semantics that are homogeneous with those of ISO14649. Within the system it is therefore possible to generate various encodings of STEP compliant information and retrieve information from various encodings. Figure 8.7 show the generation of ISO10303-21 encoding of the manufacturing information for a simple part generated by the interoperable CAD/CAM system. Due to the problems mentioned in 3.3.2 with the availability of suitable XML schemas, the prototype interoperable network does not support encoding STEP-NC data in XML format. With the finalisation of the XML encoding standard for STEP it will be possible to add the functionality

in IP^{3}AC. All systems developed based on IP^{3}AC including the interoperable CAD/CAM system can immediately utilise the new encoding as soon as it is incorporated in the IP^{3}AC foundation.

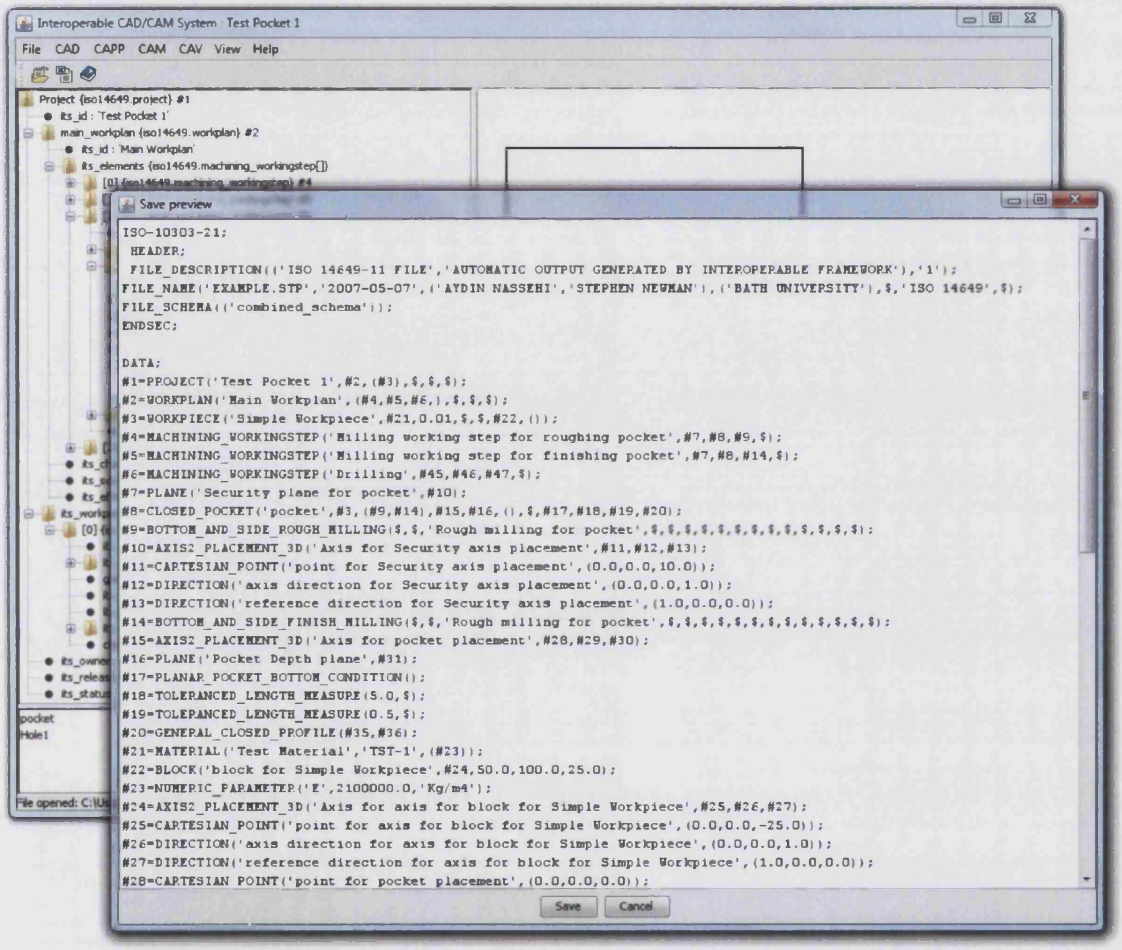


Figure 8.7 - STEP-NC generated in the interoperable CAD/CAM system

The interoperable CAD/CAM system shows an example of direct resource abstraction as mentioned in section 5.3. The system allows the user to manipulate manufacturing information using a CAD/CAM interface. It also allows the user to store and retrieve data using the chosen manufacturing Interlingua (i.e. STEP-NC) without any requirement for translations or semantic transformations. The overall functional diagram of the Interoperable CAD/CAM system can be seen in figure 8.8.

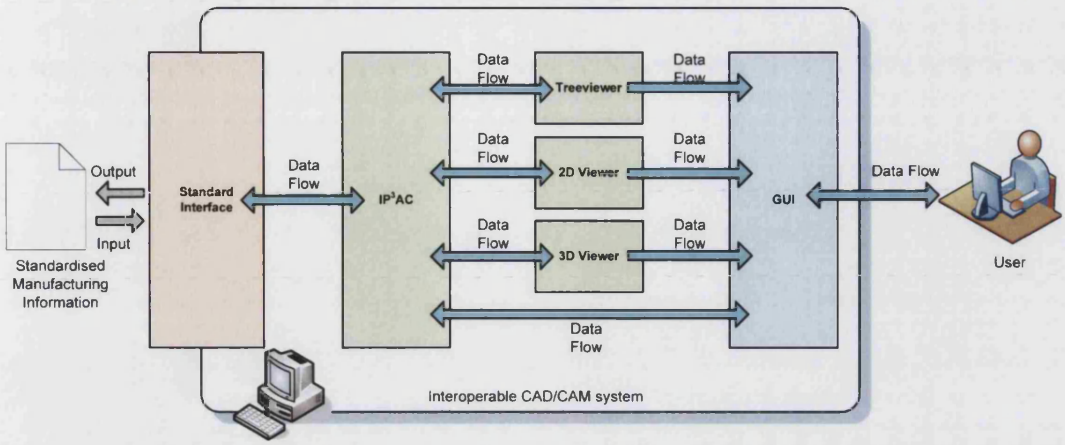


Figure 8.8 - Overall functional view of the Interoperable CAD/CAM system

8.3.2. Resource Abstraction for the Siemens 840D CNC

The Shopmill shopfloor programming system from Siemens allows the user to utilise a feature based programming interface to quickly design and create process plans for prismatic parts. Figure 8.9 shows the Shopmill interface on a Siemens 840D CNC controller.

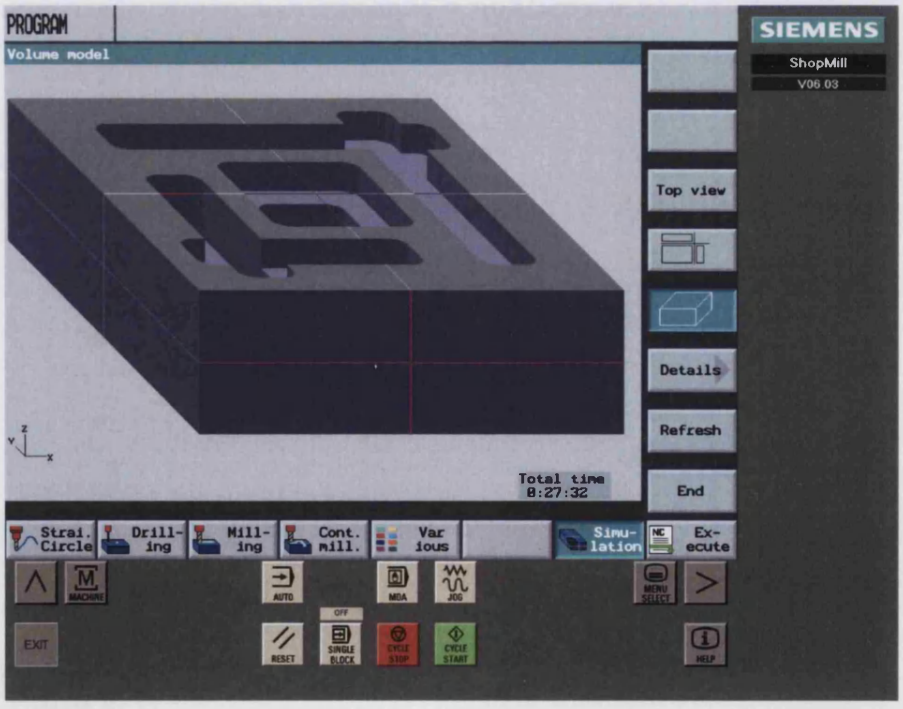


Figure 8.9 - ShopMill user interface

ShopMill provides a level of abstraction on the CNC by hiding the G&M codes from the user and only allowing modifications on a feature based level. ShopMill information is encoded as a text file with an extension of .MPF. The semantics used in ShopMill are very similar to those in ISO14649 and therefore creating an abstraction object for this controller involves minor semantic adjustments. Table 8.3 lists ShopMill features with their equivalent STEP-NC constructs as implemented in the prototype.

Table 8.3 - ShopMill features and equivalent STEP-NC entities

ShopMill Feature	STEP-NC equivalent	Description
E_HEAD	Header Section, workpiece	The header of the file, includes the dimensions of the workpiece and its shape.
E_MI_PL	plane_milling, planar_region	Describes a facing operation where a plane is machined on the part
E_CON	composite_curve	Describes a composite curve formed of line and arc segments
E_CP_CO	bottom_and_side_milling, closed_pocket	The operation required to machine a closed pocket whose boundary is described by and E_CON. Bosses can also be specified using E_CON curves.
E_PO_CIR	circular_closed_profile, bottom_and_side_milling, closed_pocket	The operation and the feature to machine a circular pocket
E_PO_REC	rectangular_closed_profile, bottom_and_side_milling, closed_pocket	The operation and the feature to machine a rectangular pocket
E_DR	round_hole, drilling	The operation and the feature to machine a round hole
E_PS_SEQ	no direct equivalent	A specified sequence of points that can be used to place one of the features
E_PS_ROW	no direct equivalent	A row of points that can be used to place one of the features

The resource abstractor for the Siemens 840D CNC has been implemented so that it will allow bi-directional information transfer, it can read ShopMill formatted data and generate STEP-NC data and vice versa. The overall view of the ShopMill abstractor can be seen in Figure

8.10. As the Siemens 840D CNC has a Microsoft Windows based front-end running on a normal PC, it is possible to install the abstractor on the CNC itself.

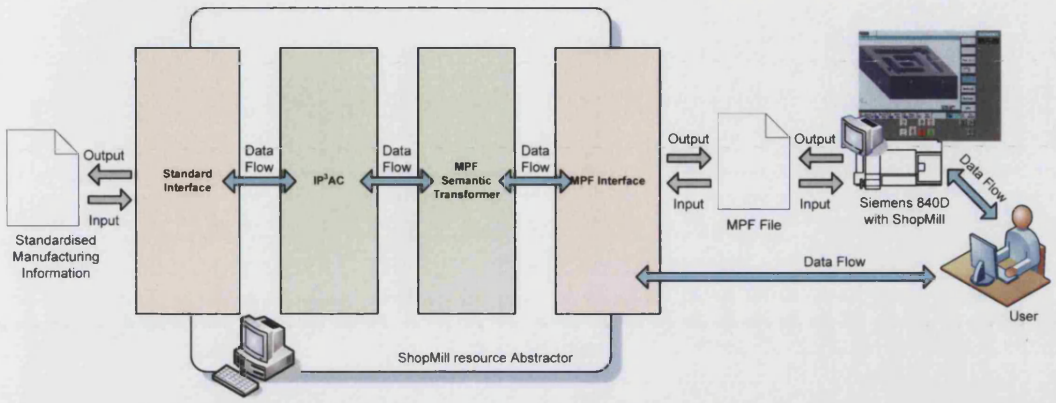


Figure 8.10 - Overall view of the ShopMill Abstractor

8.3.3. Resource Abstraction for the Heidenhain iTNC530 CNC

The Heidenhain iTNC530 CNC offers another feature based shop floor programming interface called smarT.NC. The interface can be seen in figure 8.11. However this interface is not a programming environment and features are translated to proprietary G&M Code cycles as shown in figure 8.12. Thus the semantics used in the programming of the Heidenhain iTNC530 CNC have more difference with STEP-NC than those utilised by Siemens.

Furthermore whilst the controller is PC based, it utilised a proprietary operating system which would not allow the Java virtual machine to run. It is therefore necessary to employ an interface PC to exchange information with the Heidenhain controller.

Since the semantic transformations involved in the bi-directional information transfer are well beyond the scope of this research (as mentioned in section 5.4) only a uni-directional abstraction object has been developed as part of the interoperable CAx framework prototype. The overall view of the resource abstractor implementation can be seen in figure 8.13.

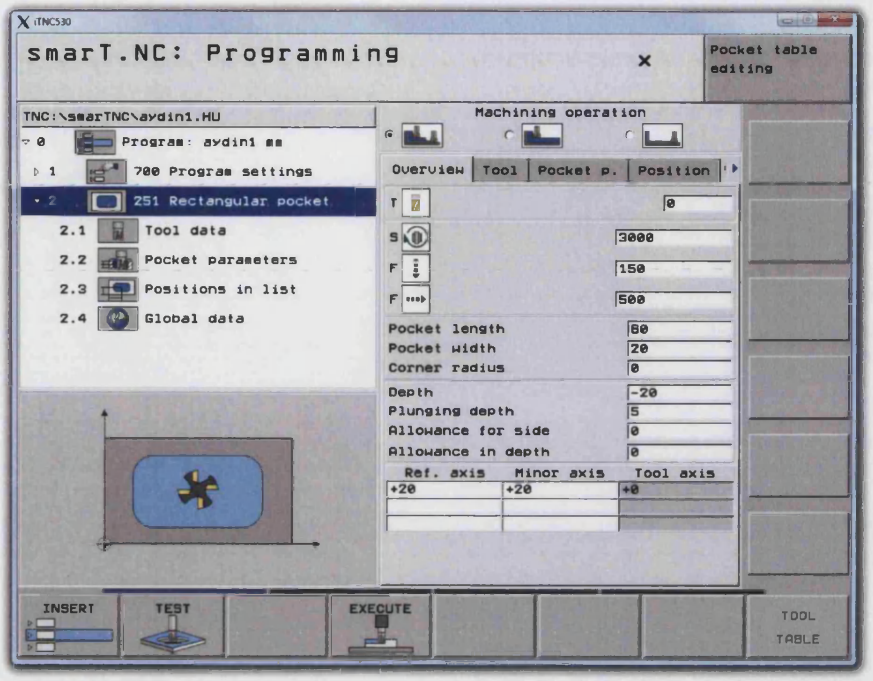


Figure 8.11- The Heidenhain iTNC530 smarT.NC programming interface

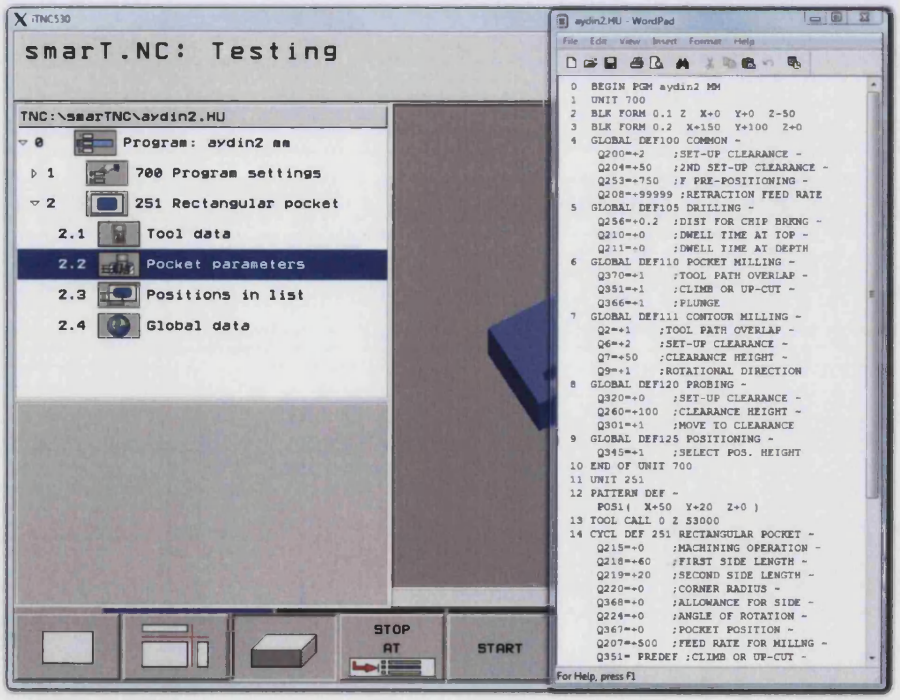


Figure 8.12 - Heidenhain proprietary G&M Code cycles

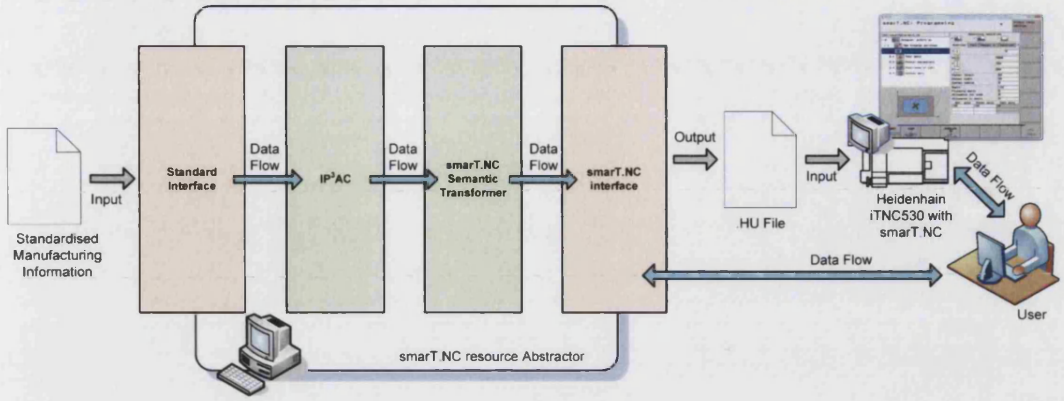


Figure 8.13 - The overall view of the Heidenhain iTNC530 CNC resource abstractor

8.4. Development of Information Transfer Procedure

In the prototype interoperable framework the number of resources and their proximity do not justify employing the agent based communication framework. As one computer is utilised to host the CAD/CAM system and connect to the CNC machines, security and reliability issues are negligible. As a result a simpler object messaging scheme was devised to encompass the logic contained within the agents and allow the various components of the prototype to exchange information with each other. This can be seen in Figure 8.14.

Furthermore it should be noted that the abstractor objects require minimal interaction with the user and as such a separate graphical user interfaces has not been developed for them in the prototype. They are controlled through the graphical user interface provided by the interoperable CAD/CAM component.

The interoperable CAD/CAM system also encompasses the functionality of the directory in the agent based framework and maintains a list of resources and their abstraction objects.

Figure 8.15 shows the interoperable CAD/CAM invoking the MPF abstraction object to create an MPF program based on the standardised STEP-NC compliant data. In this particular dialect of MPF, inspection instructions to adjust machining offsets based on the placement of the raw stock on the machining bed have also been created using standardised STEP-NC inspection routines.

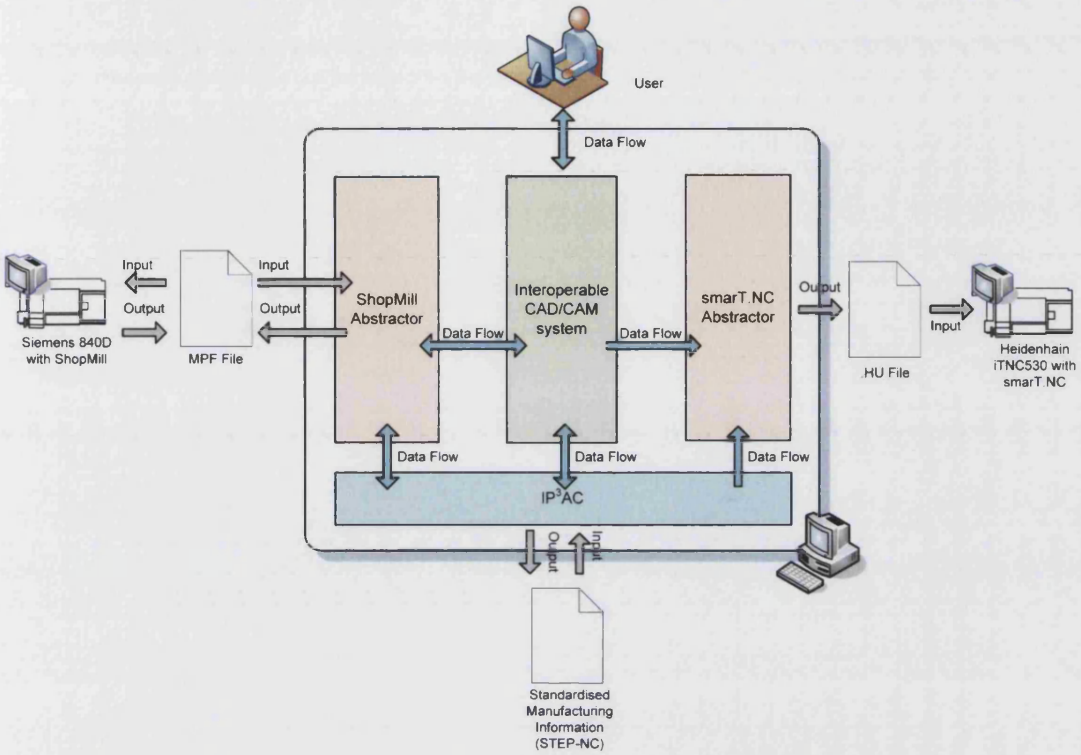


Figure 8.14 - Information transfer overview in the interoperable framework prototype

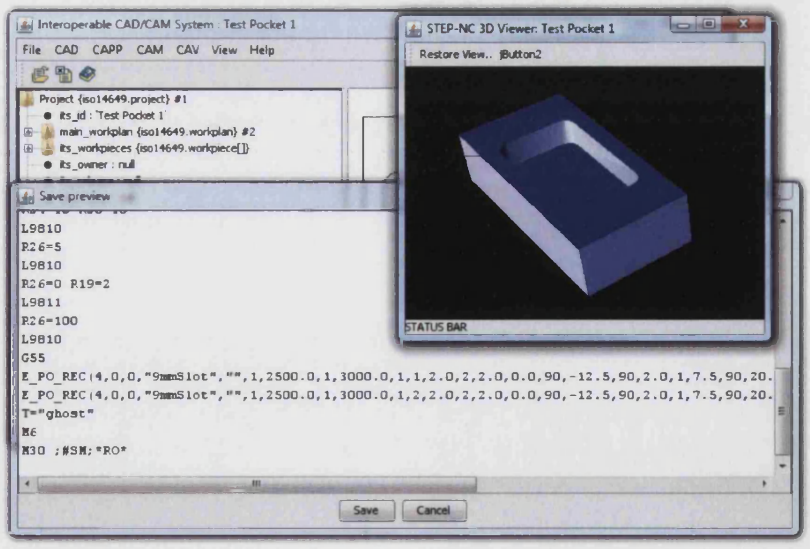


Figure 8.15 - MPF abstraction object invoked through the interoperable CAD/CAM system

8.5. Prototype Integration

Using the information transfer method from 8.4 the various components of the interoperable framework prototype have been integrated to function as a single system. Figure 8.16 shows the class diagram for the major components of the interoperable CAx system.

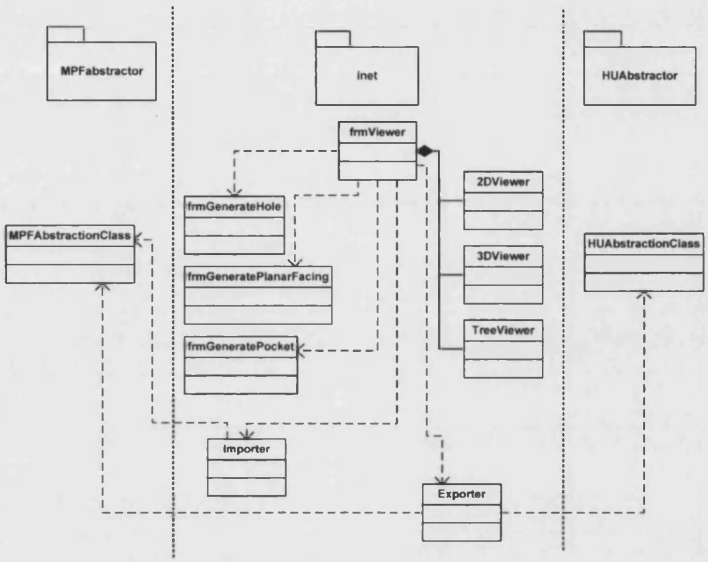


Figure 8.16 - Key classes of the interoperable framework prototype

The *frmViewer* class is the main user interface that has the other viewers as its components. It creates objects from the other *frm* classes to create IP^{3}AC objects in the Java object space. With the command of the user it creates the importer and exporter objects who instantiate the appropriate abstraction classes to handle the knowledge transformation.

8.6. Summary

In this chapter a prototype implementation of the interoperable CAx framework has been realised. The prototype utilises three CAD/CAM/CNC resources to provide a demonstration platform for the interoperable framework. The development of the prototype has also highlighted the issues facing system integrators in adopting the interoperable CAx framework. The prototype can be used in conjunction with manufacturing scenarios to test the feasibility of the interoperable CAx framework for prismatic part manufacturing.

9. Evaluation of the Interoperable CAx Framework Prototype

9.1. Introduction

In order to assess the applicability of the interoperable framework an example part has been utilised to illustrate manufacturing scenarios within the limitations of the prototype. The example part has been designed based on a number of similar components utilised within the aerospace industries. In the designated manufacturing scenario, the part is programmed and machined on a Siemens 840D CNC, the data is then transferred and edited on the interoperable CAD/CAM system and finally transferred to a Heidenhain iTNC530 CNC for machining.

9.2. Evaluation Methodology

In order to evaluate the prototype implementation of the interoperable CAx framework, a manufacturing scenario has been defined. In this manufacturing scenario, a test component is required to be produced on two different machines with two different controllers as seen in figure 9.1. The part programme for the component is created at one of the CNCs using a shopfloor programming system.

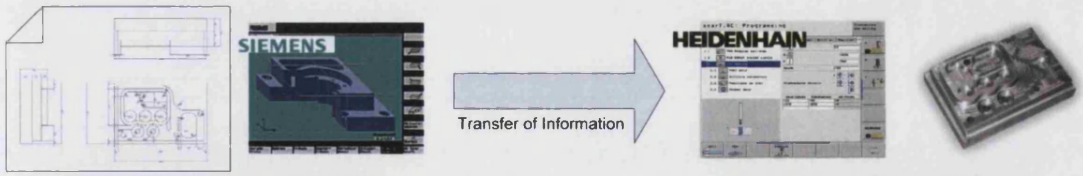


Figure 9.1 - The manufacturing scenario for evaluation of the prototype

In the current CAD/CAM/CNC chain, an engineer will need to re-evaluate the process plan in the CAM system and then postprocess it for the second CNC-machine. Alternatively the component might be reprogrammed on the second CNC-machine using a shopfloor programming system.

Using the interoperable CAx framework prototype this scenario is handled automatically. The following sections document the various steps in realisation of the test scenario using the prototype implementation of the interoperable CAx framework.

9.3. Test Component Design

The example part illustrated in figure 9.2 was chosen to prove the functionality of the interoperable framework prototype. This part is designed based on a number of components used in the aerospace industry, but modified to eliminate intellectual property issues.

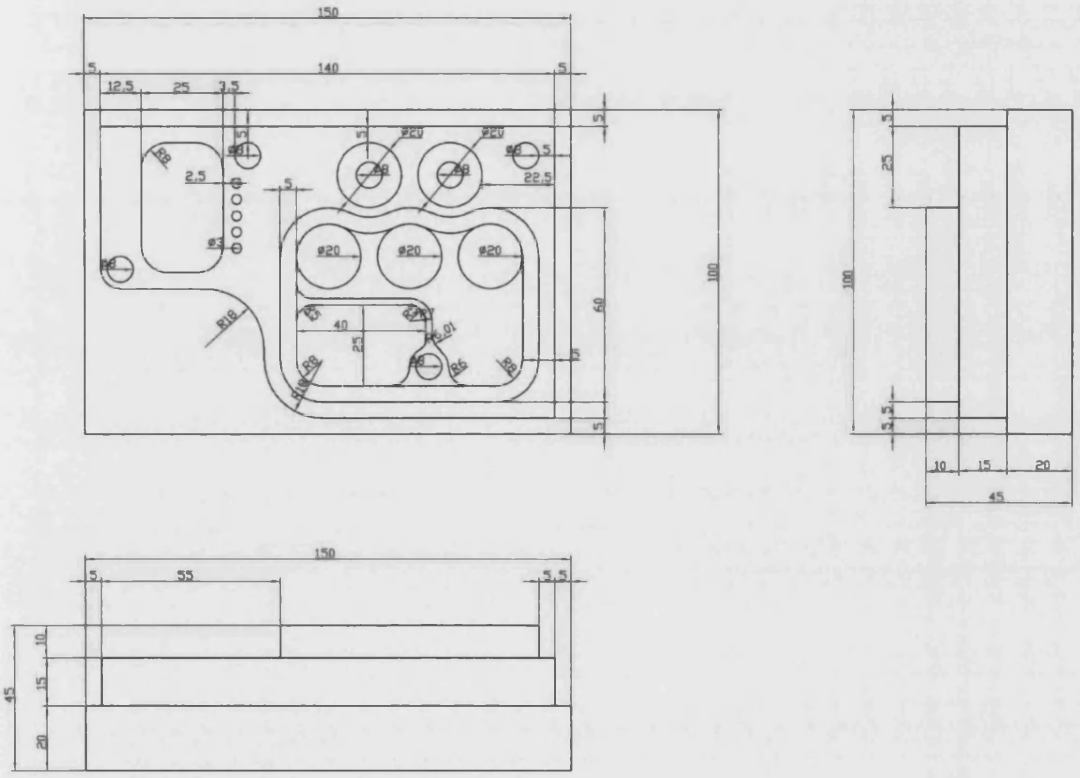


Figure 9.2 - Test component drawing

The test component comprises a number of manufacturing features to be cut from a 150mm x 100mm x 50mm block. To evaluate the interoperable CAx framework prototype these features and relevant operations are programmed on one CNC machine. The part programme is then transferred to a second CNC machine with a different controller. In this transfer the semantics of the process plan remain unchanged while the syntax used to express the manufacturing information is altered significantly.

The features and the operations that make up the process plan are as follows:

(i)Planar facing: The preparation of the test component begins by a facing operation that removes 5mms from the top of the stock as illustrated in figure 9.3.

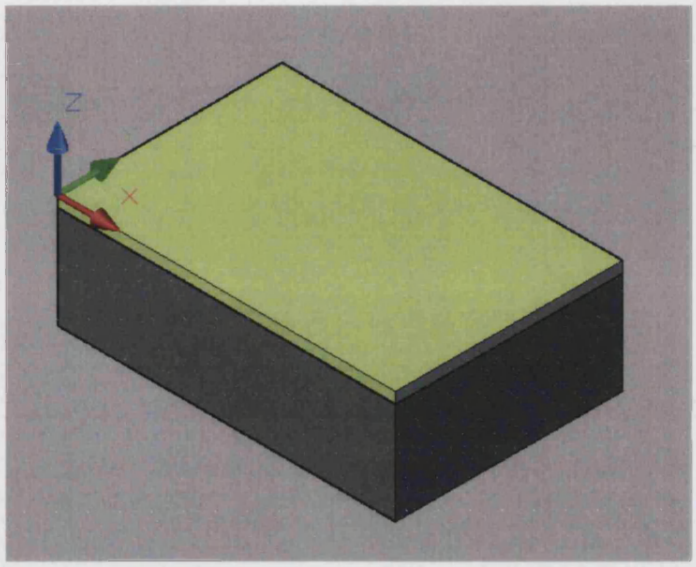


Figure 9.3 - Planar facing operation

(ii) Outermost pocket and Boss: To create the boss in the test component the pocket and the island in figure 9.4 are machined.

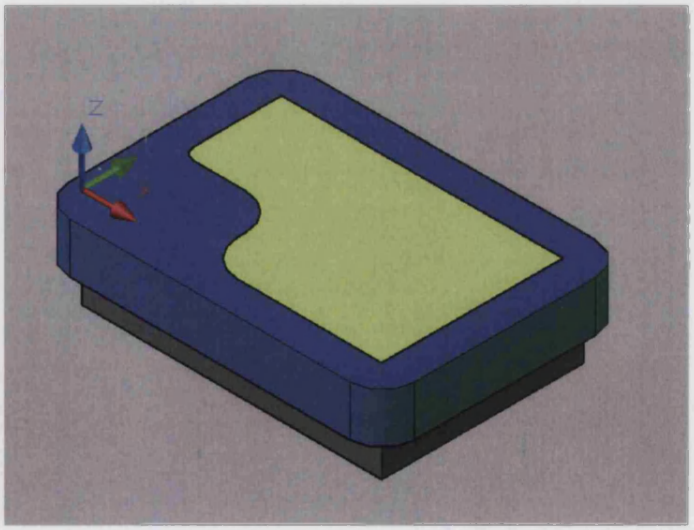


Figure 9.4 - Outermost pocket machining with boss

(iii) Outer pocket and boss: another pocket and island are defined to create the boss on top of the part as seen in figure 9.5.

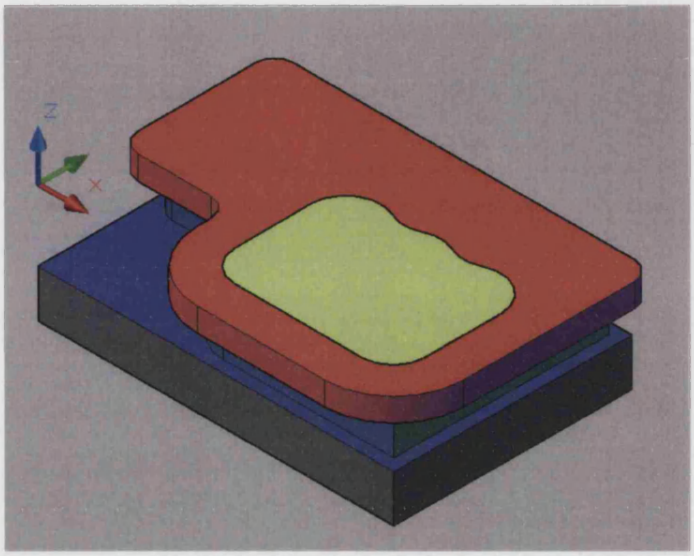


Figure 9.5 - Outer pocket machining with boss

(iv) Inner pocket: the yellow pocket in figure 9.6 is machined.

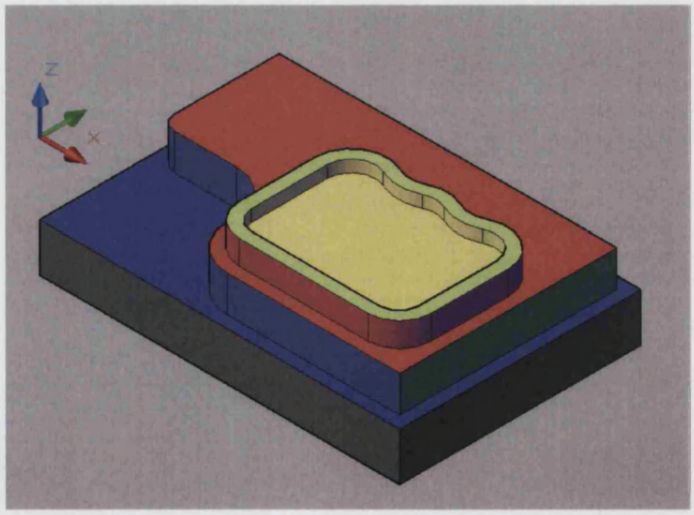


Figure 9.6 - Inner pocket machining

(v) The second tier pockets: these pockets are machined as highlighted by magenta in figure 9.7 and cyan in figure 9.8.

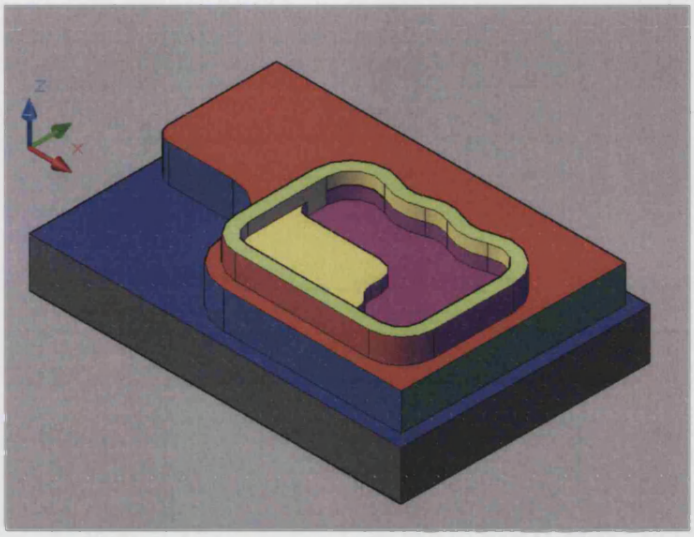


Figure 9.7 - Second tier pocket 1 machining

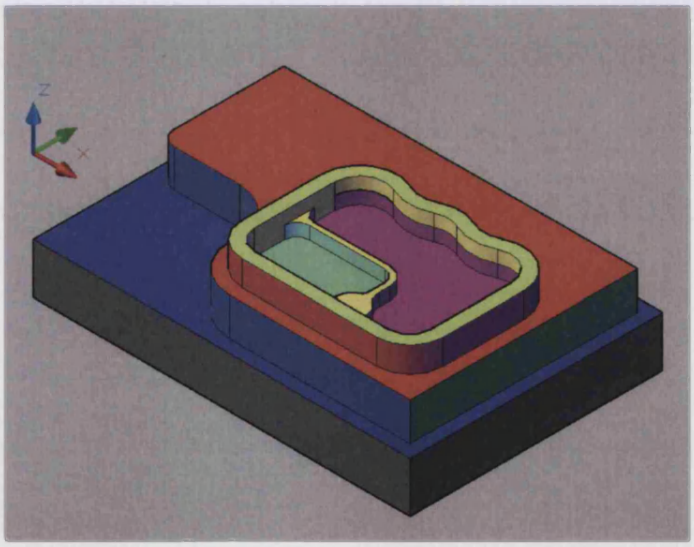


Figure 9.8 - Second tier pocket 2 machining

(vi) Circular pockets: Five circular pockets are machined on top of the part as seen in figure 9.9.

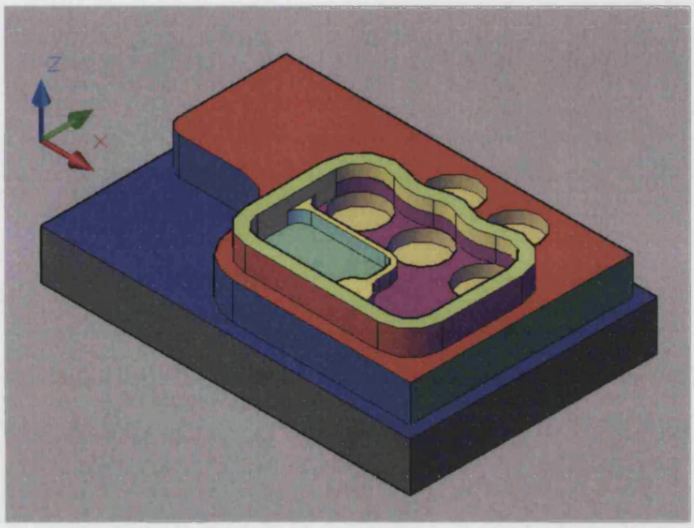


Figure 9.9 - Circular pockets machining

(vii) Rectangular pocket: The rectangular pocket is machined as highlighted by magenta in figure 9.10.

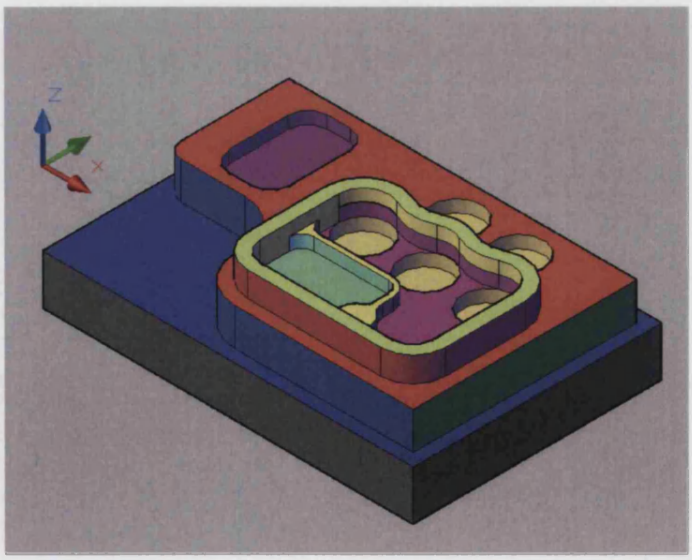


Figure 9.10 - Rectangular pocket machining

(viii) 8mm holes: six holes with a diameter of 8mms are drilled on the part as seen in figure 9.11.

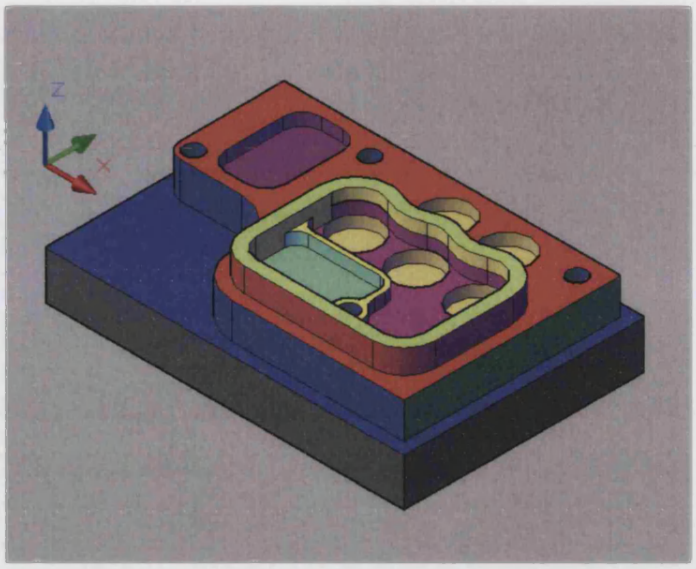


Figure 9.11 - 8mm holes drilling

(ix) 3mm holes: five 3mm holes are drilled on the part as seen in figure 9.12.

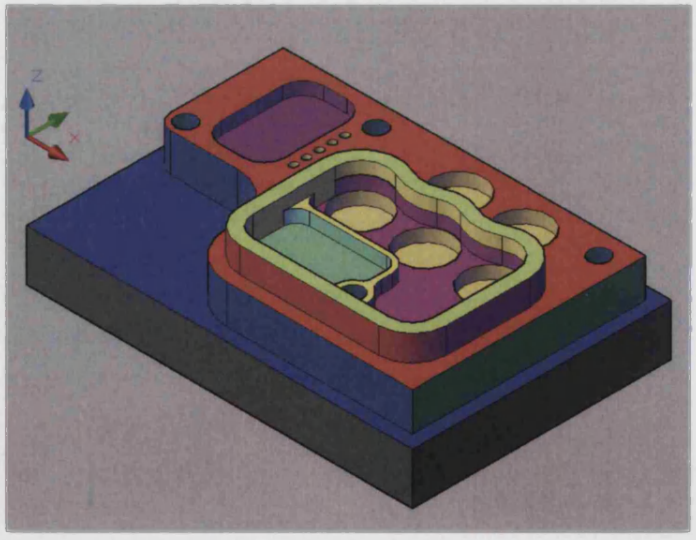


Figure 9.12 - 3mm holes

9.4. Test Component Programming On CNC 1 with Siemens 840D Controller

The process plan in 9.3 was created using the ShopMill interface on the 840D controller. As tools need to be specified in ShopMill, a 16mm Slot Drill, a 63mm Face Mill and two drills with diameters of 3mm and 8mm were defined. An excerpt of the ShopMill code can be seen in figure 9.13.

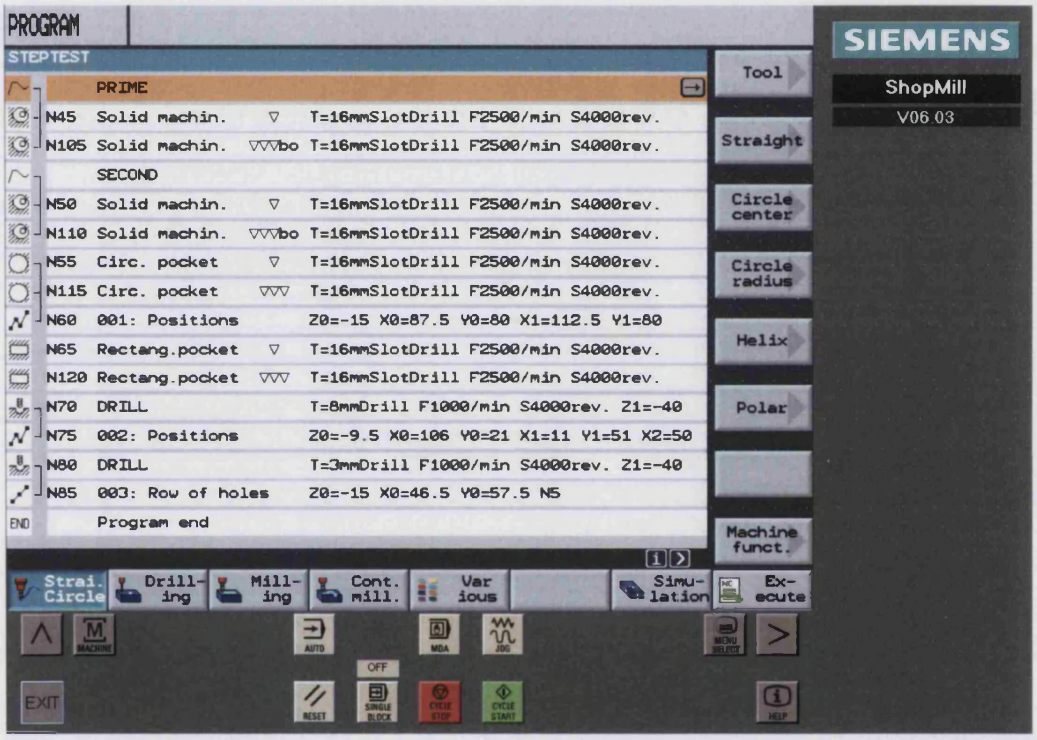


Figure 9.13 - ShopMill code excerpt

The part was then simulated using the internal simulation engine provided by the Siemens 840D controller. The results of the simulation can be seen in figure 9.14 as two dimensional drawings and figure 9.15 as a three dimensional representation. The part was then machined using a Bridgeport VMC610 equipped with the 840D controller as seen in figure 9.16.

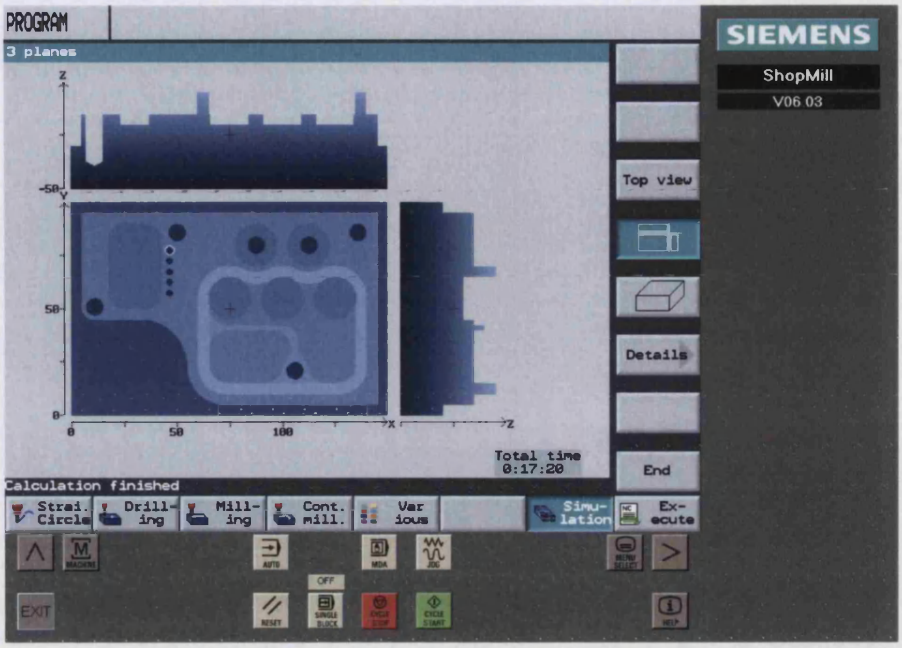


Figure 9.14 - Two dimensional ShopMill simulation

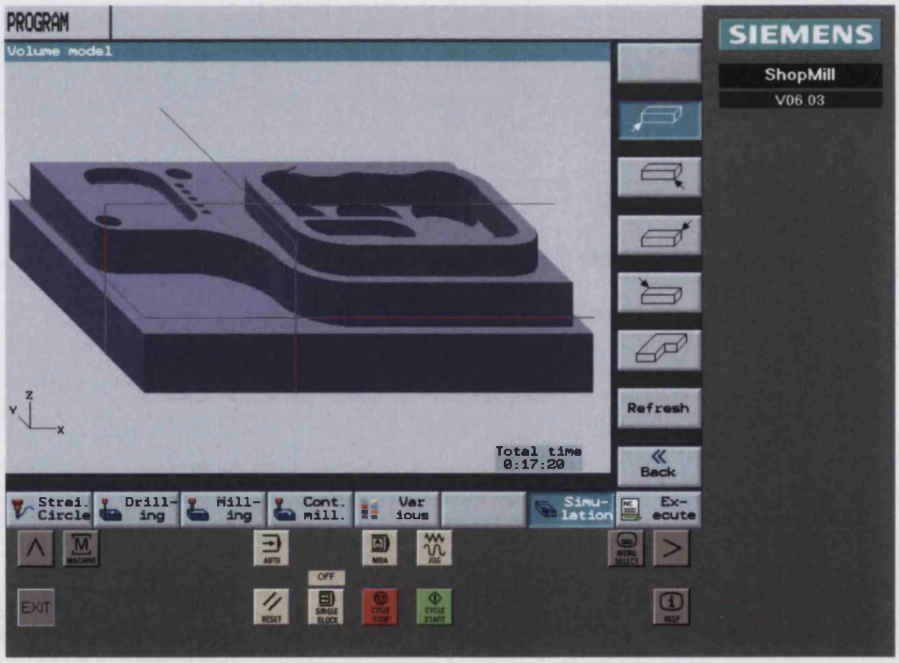


Figure 9.15 - Three dimensional ShopMill simulation

Figure 9.16 - Machined part

9.5. Generation of the Standardised Manufacturing Information

Upon the completion of the first stage of the test where the part was programmed and machined on the first CAx resource, the interoperable CAD/CAM solution was launched and the ShopMill abstraction interface invoked. The interface generated the STEP-NC representation of the process plan as seen in figure 9.17. As the semantics in the ShopMill controller are slightly different to those in the STEP-NC process plan, a number of transformations were necessary. The hole positions for example were defined using individual hole placements in the STEP-NC. While STEP-NC does provide entities to model a series of holes, in the prototype each hole is defined with its own workingstep. The generated STEP-NC workplan and the geometry as visualized by the two dimensional viewer component of the interoperable CAD/CAM system can be seen in figure 9.17.

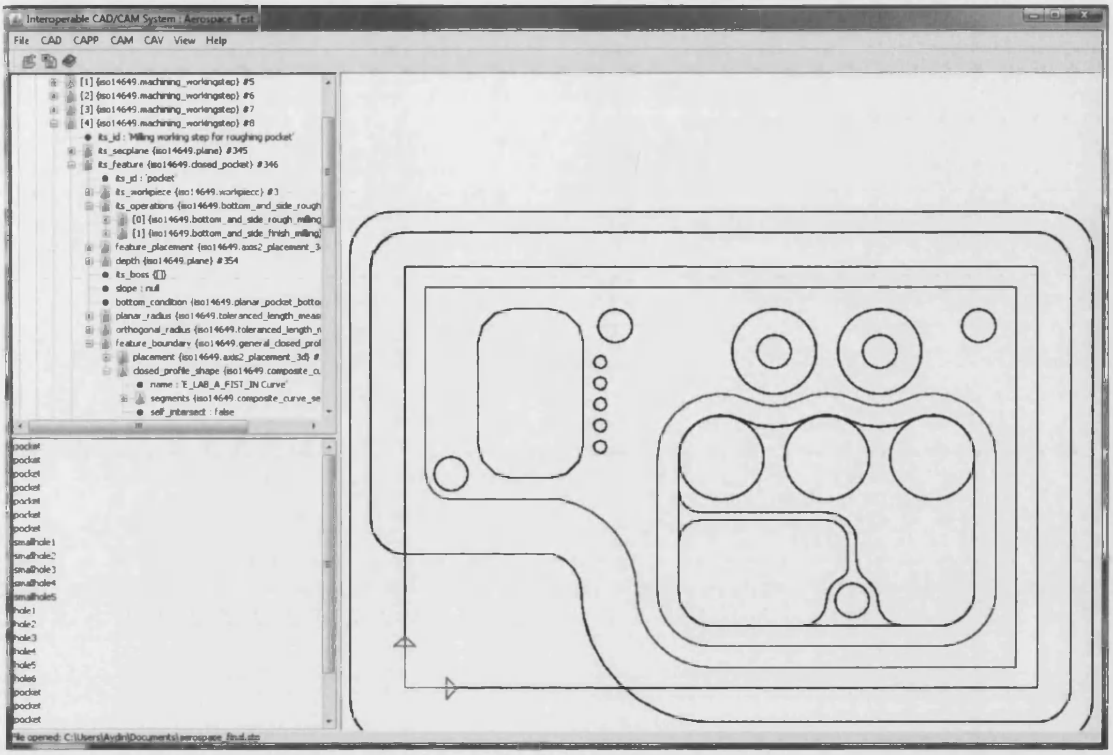


Figure 9.17 - The test component in the interoperable CAD/CAM system

9.6. Generation of Proprietary Code for Heidenhain iTNC530

The interoperable CAD/CAM system then passed the data on to the Heidenhain abstractor that generates the suitable cycles for an iTNC530 controller. Figure 9.18 shows an excerpt of the generated cycles.

The smarT.NC shopfloor programming system on the iTNC 530 provides a graphical front-end for the programming cycles. Figure 9.19 shows the test component simulated using the generated cycles for the Heidenhain controller.

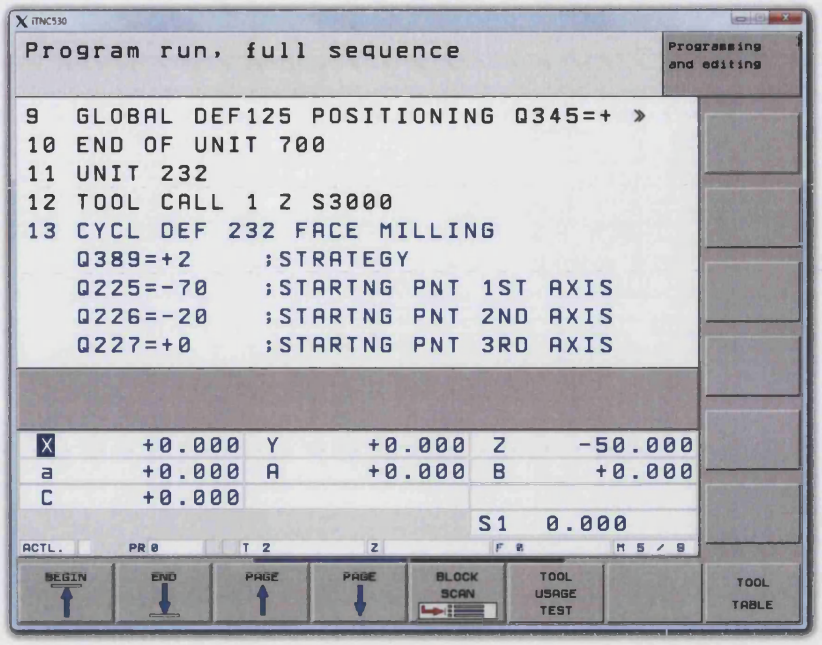


Figure 9.18 - Heidenhain cycles

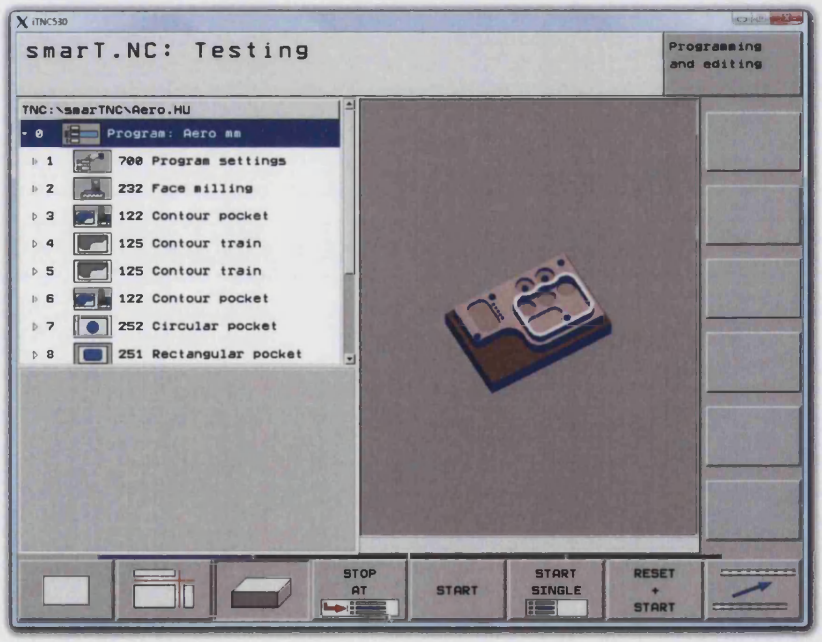


Figure 9.19 - Heidenhain simulation of the test part

9.7. Results of the Evaluation

With the state-of-the-art the engineering effort has to be applied twice to achieve the goal of the scenario. This is shown in figure 9.20.

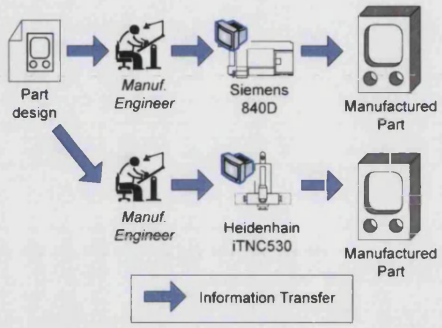


Figure 9.20 - Manufacturing scenario using state-of-the-art

A comparison of the STEP-NC process plan, the ShopMill process plan and the Heidenhain program (see appendix B for program listings) generated by the interoperable CAx framework prototype for the test component shows that the semantics of the original process plan as defined on the ShopMill systems has been preserved during the transfer of data. Using the prototype, therefore, it is possible to achieve the goals of the scenario with one engineer as seen in figure 9.21. With the automatic nature of data transfers, the two CNC machines can be used interchangeably adding to the resource flexibility of the enterprise.

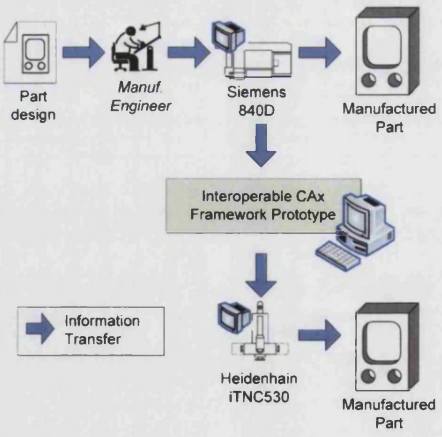


Figure 9.21 - Manufacturing scenario using the interoperable CAx framework prototype

10. Discussion

10.1. Introduction

In this chapter a number of issues that have been considered in the course of the research in relation to the research scope and context are discussed.

10.2. State-Of-The-Art in CAD/CAM/CNC Integration

The review of the literature on information integration in the current CAx chain, identified in chapter 3, has shown that the current chain fails to support the new requirements of users for flexibility and interoperability. In the current post-processor based CAx chain, interoperability is vendor specific. In order to enable data transfer between any two specific CAx resources, a translator is required. This multi-syntax, multi-semantic environment makes it impractical to develop translators for every single combination of CAx resources. The lack of interoperability leads to diminishing resource fluidity and ultimately loss of business and market share for manufacturing enterprises.

The standardisation effort that has begun in the form of STEP-NC has, so far, failed to attract commercial support. The standard is being perceived by CAx developers as a threat to their established user bases. In addition, fundamental changes in the CAx resources are necessary to make them STEP-NC compliant. It is also noteworthy that the foundations of STEP were designed when using procedural languages and structured programming were still the prominent software development paradigm. As identified in 3.3.2, with the worldwide paradigm shift from structured programming to object oriented programming, the development of future generations of STEP - and consequently STEP-NC - compliant CAx resources, requires an awkward fusion of two mentalities from two eras.

The author is also of the opinion that a large proportion of the previous research into interoperability of CAx systems has been conducted on a reactive basis. As a result the author believes, a re-imagination of the CAD/CAM/CNC domain based on new paradigms and technologies is necessary. Proactive research into interoperability and creating an extensible

framework for CAx interoperability can not only solve today's issues but also provide the necessary tools to solve future problems as well.

10.3. A Novel Framework for Realisation of Interoperability

The framework envisioned in chapter 4, provides a novel and extensible foundation for enabling interoperability among CAx resources. The philosophy behind the framework is increasing information availability and semantic homogenisation.

By increasing information availability, every bit of knowledge that is generated during the CAx manufacturing process is captured. Semantic homogenisation ensures that the captured information maintains the original intended meaning.

This is an exceptional improvement over the information handling in the current CAx chain, where bits of information are lost along the process chain (i.e. geometry is not passed on to the CNC controller and G&M codes file contains nothing but axis movements). Through the use of a homogenised set of semantics, the framework enables every CAx resource to exchange information with others through the use of a single interface. The framework is also fundamentally different to the standard based approach where all resources need to adopt a single set of semantics together with a specific syntax to allow information exchange. Since CAx resources can be utilised and integrated without major internal modifications, the interoperable CAx framework offers a considerable commercial advantage.

The framework offers three functionalities:

- Resource abstraction where the syntax of CAx resource data is analysed and manufacturing semantics are derived
- Encoding manufacturing information to store and retrieve information based on the Interlingua chosen for the homogenised semantics, and
- A communication mechanism to ensure the integrity of information during transfer.

10.3.1. Manufacturing Resource Abstraction

As identified above, a pivotal element for realising the interoperable framework is the abstraction of resources. Managing information exchanges in a heterogeneous environment where a plethora of resources communicate their distinctive semantics with various syntaxes is an exceedingly difficult task to accomplish. Resource abstraction homogenises such an environment by allowing different systems to communicate using a subset of the concepts within a single lexicon.

In chapter 5 of this research, an innovative resource abstraction method for CAx resources was presented. The method covers resources that utilise the subset of CNC manufacturing lexicon that covers the production of prismatic components. The extensible approach for resource abstraction relies on XML definitions of resources to configure syntax translation and semantic transformation. It is thus possible for CAx vendors, framework developers or end-users to define XML definitions and hence allow a specific CAx resource to be abstracted and connected to the framework without the need for additional programming.

The abstraction process in this research empowers manufacturing enterprises as well as CAx vendors to move towards a "plug-and-manufacture" paradigm. The widespread popularity of XML and the user-editable definitions means that everyone can define the resource parameters. It would be possible for vendors to store the XML definition on the resources themselves and hence as soon as the resource is connected to the network, it is interoperable with all existing CAx resources already on the network.

10.3.2. Encoding Manufacturing Information

The semantically homogenised information obtained from resource abstraction is encoded according to a comprehensive manufacturing lexicon. The computational platform titled IP^{3}AC, described in chapter 6 of this research, utilises the information structures provided by STEP-NC for prismatic parts to encode the manufacturing information.

IP^{3}AC extends the STEP-NC information models and transforms them into the native object space of the Java programming language. This approach circumvents the problems described in 10.2 in fusing STEP oriented standards with object oriented languages. Instead of utilising

SDAI, the native STEP data access interface (see section 3.3.2), the object based model of IP^{3}AC allows data structures to be handled natively in the programming environment.

In addition to this fundamental advantage, the object oriented approach in IP^{3}AC enables programmers to utilise existing Java libraries for visualisation, simulation, calculation, etc. in conjunction with the manufacturing information encoded by IP^{3}AC with minimal effort.

10.3.3. Utilisation of Mobile Agents to Provide Communications

An unregulated transfer of data between CAx resources could adversely affect the integrity of the manufacturing information. In chapter 7 of this research, a mobile agent based communication system has been designed to regulate and facilitate information exchange throughout the CAx interoperable framework.

The use of mobile agents in manufacturing has a number of proponents and opponents. While the critics cite problems such as asynchronised flow of execution, difficulty of monitoring individual agents throughout the network and unproven reliability in converging on a single solution, the benefits presented in chapter 7, make the use of mobile agents an attractive method for transferring manufacturing information in a global enterprise.

In such an environment, the network connections are not completely reliable, security of data transfer is of utmost importance and any misinterpretation of information could be catastrophic from a business point of view. Mobile agents with their inherent security and their capability to maintain a state of execution in addition to data can ensure integrity and correct interpretation of information. The robust migration techniques utilised by mobile agents also means that transfers are reliable despite unreliable network infrastructures.

10.3.4. Prototype Implementation

In order to demonstrate the interoperable CAx framework, a prototype has been implemented. As described in chapter 8, this prototype is realised using three CAx resources; two commercial CNCs: the Siemens 840D and the Heidenhain iTNC530 and a prismatic CAD/CAM system fully developed in Java to demonstrate direct abstraction.

The prototype allows data to be bi-directionally transferred between the Siemens controller and the STEP-NC CAD/CAM system. The integrity of the information is ensured and the semantics are homogenised during the transfer. The prototype also allows manufacturing information to be sent from the CAD/CAM system to the Heidenhain controller.

While the abstraction and encoding elements of the framework have been fully implemented in the prototype, a scaled down version of the communications system has been chosen for implementation. This has been due to the fact that in the prototype, the information is stored locally and the additional overhead of an agent based system is not justifiable. The same object oriented logic for message exchanges, which is contained within the agents, however, has been maintained during the development of the prototype.

The prototype provides an actual platform to demonstrate the advantages of realising interoperability in the CAx resource chain. Transferring information among the three resources integrated in the prototype highlights the brilliance of homogenising the similar semantics of the resources to get around the usual syntax translation problems of post-processors.

10.4. Evaluation of the Prototype Using a Test Component

To evaluate the prototype implementation of the interoperable CAx framework, an industrially inspired prismatic component has been utilised in a typical manufacturing scenario. The scenario entails the design of the component on the ShopMill shopfloor programming system on a Siemens CNC and then manufacturing it on a Heidenhain CNC.

In the current CAD/CAM/CNC chain, there are no contingencies for taking information back from a shop floor programming system (except perhaps, in the special case of Mazak where the entire manufacturing information is unified using proprietary information models that are not useful for the controllers in the specified scenario). An engineer will therefore need to re-evaluate and possibly re-create a process plan in the CAM system and then use a post processor to generate the part programme to manufacture the part on the Heidenhain controller.

Using the prototype, the process plan is directly transferred from the Siemens controller to the CAD/CAM system and from there to the Heidenhain controller with no requirement for redefining the data. Semantics contained within the process plan including the geometry, the operations and the tooling and feeds and speeds are all maintained throughout the transfer.

10.5. Advantages of Implementation of the Interoperable CAx Framework

This paradigm shift from a chain of CAx resources to an interoperable network for CAx resources is illustrated in figure 10.1. In an interoperable network, instead of unidirectional information transfer in a single chain of CAD, CAM and CNC, the manufacturing data can be exchanged freely between the various resources.

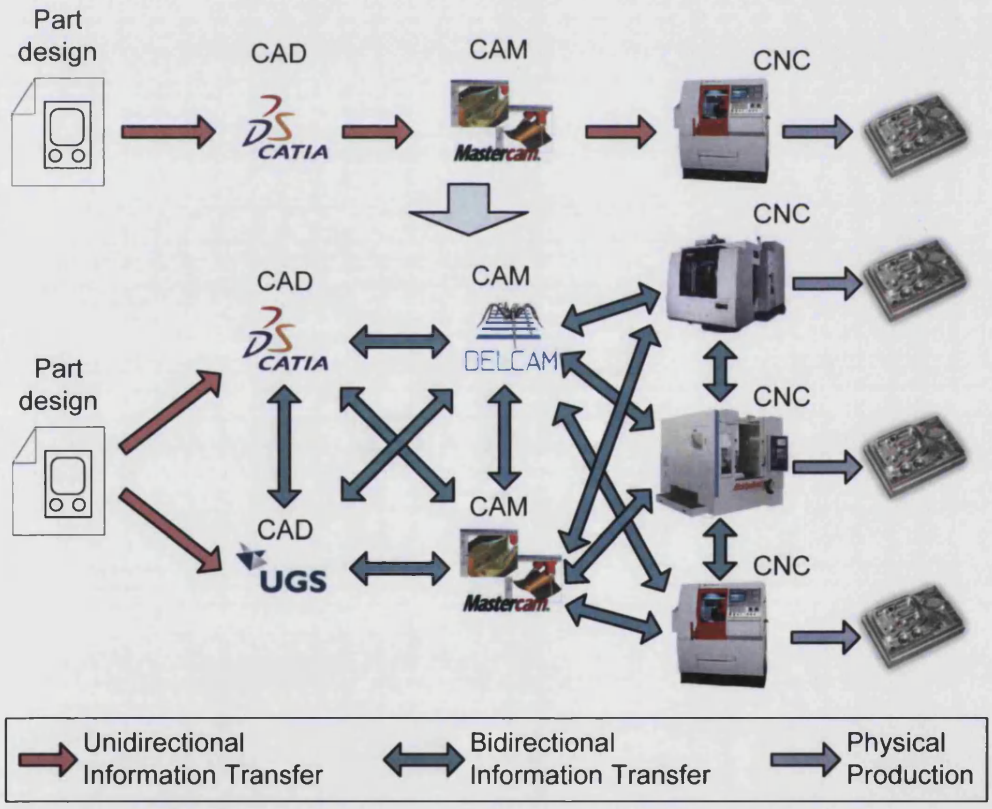


Figure 10.1 - Paradigm shift from CAx chain to interoperable CAx network

Realisation of the interoperable CAx network will enable a global manufacturing enterprise to employ a "design anywhere, manufacture anywhere" approach and hence invigorate its

presence and responsiveness in local and global markets. Furthermore, as the integrity of the information transfers can be ensured, resources in various enterprises can form a resource pool for a conglomerate virtual enterprise, enabling a group of small to medium manufacturing enterprises (SMEs) to undertake large manufacturing jobs.

The advantages of using the interoperable CAx framework for enabling interoperability in manufacturing enterprises are evident in Table 10.1, where the various integration schemas are discussed from the perspective of the various players involved.

Table 10.1 - Comparison of the various interoperability approaches

	State-of-the-art	STEP-NC	Interoperable CAx framework
End Users	No solution for universal interoperability. Users require a translator for transferring data between each two CAx systems.	Need to adopt a feature based, STEP compliant design and manufacturing mentality. The entire CAx network is standardised.	Can utilise the existing knowledge and current resources without the requirement to change methods. Future standardised CAx resource can also be integrated,
CAD Vendors	Vendors need to develop CAM interfaces for industry leading CAM systems as well as those requested by customers or face losing business.	A fundamental overhaul of software is required to make it standards compliant. Might be perceived as losing market advantage	The software can be marketed in its current form. With the provision of an XML abstraction file, the CAD system can communicate with every existing resource within the framework.
CAM Vendors	Vendors need to develop CAM interfaces for industry leading CAD systems as well as those requested by customers or face losing business. Post-processors for various CNC machines should also be developed.	A fundamental overhaul of software is required to make it standards compliant. Might be perceived as losing market advantage	The software can be marketed in its current form With the provision of an XML abstraction file, the CAM system can communicate with every existing resource within the framework
CNC Manufacturers	Vendors need to develop post-processors for industry leading CAM systems or face losing business.	A fundamental overhaul of controller software is required to make it standards compliant. Might be perceived as losing market advantage	The software/hardware can be marketed in its current form. With the provision of an XML abstraction file, the CNC can communicate with every existing resource within the framework
Paradigm	**Many Formats, Many Meanings**	**One Format, One Meaning,**	**Many Formats, One Meaning**

The ingenuity of the interoperable CAx framework is in that it allows users to utilise their existing resources in an integrated environment while supporting future standardisation. It also allows vendors to maintain their competitiveness by not requiring them to adopt a specific interface. In addition it will allow CAx vendors to transfer information to a wider variety of CAx resources without requiring them to devise specific interfaces.

At the same time, the framework is future proof. This is due to the extensible approach to the creation of the manufacturing lexicon as well as the XML based resource abstraction. Even though the nature of the information required by future CAx resources is not known, given that the future information models can be expressed in an object oriented manner, the manufacturing lexicon can be extended to accommodate them.

With the extended lexicon, the interoperable CAx framework conveniently transforms the process of interfacing with new CAx resources, to a matter of creating suitable XML definitions of the resources. The adoption of the interoperable CAx platform is therefore advantageous to users and vendors alike and can proactively provide the foundation for the next generation of CAD/CAM/CNC research in a fully integrated environment.

10.6. Limitations of the Interoperable CAx Framework

There are a number of limitations associated with the framework proposed in this thesis.

(i) Defining the geometry of prismatic parts using features is relatively simple. Extending the feature based interoperable framework to support parts that have a more complex geometry, such as those manufactured using 5-axis milling machines, can prove to be difficult.

(ii) The ability to generate high-level information such as geometrical features based on low-level information such as tool paths is currently unproven. This presents a major challenge in developing bi-directional interfaces for legacy CNC resources within the interoperable CAx framework.

(iii) The framework, as presented in this thesis, is not capable of handling the data pertaining to geometric tolerances, different setups or fixtures. Correct handling of this information poses challenges that have to be resolved for further development of the framework.

11. Conclusions and Future Work

11.1. Introduction

In this chapter the conclusions that have been derived as the result of this research are provided, together with suggestions for future areas of investigation.

11.2. Conclusions

- In order to enable universal interoperability across the CAx manufacturing chain, the semantics of the information contained within all resources should be homogenised. Thus, the consolidation of the information contained within the resources should provide a consistent set of meanings. No contradictory or unrelated pieces of information should exist throughout the chain.

- The current solutions for integration of CAx systems are unable to support universal interoperability between the various resources. For transferring data in this "many formats, many meanings" environment from each specific resource to another, a tailor-made translator is required. As the semantics are not homogenised throughout the chain, this is sometimes difficult to achieve.

- STEP-NC has provided a way forward by proposing a "one format, one meaning" manufacturing domain. In STEP-NC all resources communicate utilising the same set of semantics and syntaxes and hence no translation or transformation is required. Adoption of the standard however requires fundamental modifications to the existing CAx resources and manufacturing methods. Despite providing excellent information models to store manufacturing knowledge, STEP-NC has failed to gather commercial following because of the above reasons and the perceived negative impacts on CAx vendor businesses.

- An interoperable CAx framework based on homogenised semantics can enable interoperability without requiring vendors to make modifications to their products. This framework can bring the benefits of the high-level information models such as those provided by STEP-NC to the state-of-the-art CAx resources without the need for

internal modification. To realise this "many formats, one meaning" framework three related components are required: (i) Abstraction to allow various resources to exchange information in a unified manner. (ii) Encoding manufacturing data in compliance with a comprehensive manufacturing lexicon. (iii) Communication to allow physical data exchange between resources.

- The abstraction method described in this research, enables the semantics of various resources to be interpreted according to the manufacturing lexicon. The use of XML definitions makes the approach extensible and allows vendors, integrators or users to create resource abstractions without programming. By including the XML file on the resource itself, a "plug-and-produce" manufacturing environment will be realised.

- STEP-NC data models have been used in this research as the basis for the manufacturing lexicon. The expertise of the standard developers has therefore been retained in creating a natively object-oriented view of the manufacturing data in the form of IP^{3}AC. This approach does not only avoid the problems associated in implementing SDAI in object oriented languages but is also substantially simpler and more versatile than SDAI in using existing programming libraries.

- Secure, robust and reliable communications for physical data transfer between resources can be provided by a mobile agent based system. In a global enterprise geographically dispersed subsidiaries are connected to each other with unreliable network links. In such an enterprise, the mobile agent based communication system can ensure the integrity of information transfer to avoid disastrous misinterpretations of data, provide reliable data transfers and guard the security of the transferred data.

- A prototype implementation of the framework connecting a CAD/CAM system with two CNC controllers has been realised to identify development issues and advantages. Modern CNC controllers and CAD/CAM systems provide similar semantics to the chosen manufacturing lexicon and therefore semantic homogenisation process is relatively simple. Legacy systems with low-level semantics on the other hand require substantial effort to integrate.

- An industrially inspired test component has been utilised in conjunction with the prototype implementation of the framework to evaluate the concept. A part program was created using the ShopMill shop floor programming system and transferred to a Heidenhain iTNC530 equipped CNC to show the seamless transfer of information enabled by the interoperable CAx platform. The automatic transformation of the data from the format suitable for one machine to the format suitable for the other machine highlights the advantages of utilising the interoperable CAx framework.

- The interoperable CAx framework presented in this thesis can be augmented to cover the wider set of technologies employed in CNC manufacturing. The ingenuity of the Interoperable CAx framework is that it provides end users of all sizes the ability to integrate their current resources as well as a versatile platform for the adoption of future generations of CAD/CAM/CNC systems. The extensible nature of the framework makes it convenient to consistently adopt future technologies as well as existing ones to create a universal manufacturing platform. In such a platform the individual CAx resources of a global enterprise are combined in a powerful, agile and reliable resource pool that can be utilised as necessary with no time or cost penalties. This provides a manufacturing enterprise with the versatility and strategic flexibility to have a highly competitive edge in the knowledge oriented market of the future.

11.3. Contributions to Knowledge

The main contribution of this thesis to knowledge is the novel vision of an interoperable framework based on homogenisation of manufacturing information semantics. The framework replaces the current paradigm of interoperability and also alleviates the problems associated with the standardisation approach that has been proposed in the form of STEP-NC. In addition the proposed open approach for resource abstraction and the suggested approach for object-oriented encoding of manufacturing information are both valuable contributions to the field.

11.4. Future Work

During the course of this research a number of opportunities for taking the work further have been identified. The extensible nature of the proposed interoperable CAx framework translates into vast potential not only for the integration of the current ongoing research in CAD/CAM/CNC but also to provide a platform for future research.

11.4.1. Integration of the STEP-NC Research

As the interoperable CAx framework uses semantics compatible with those provided with STEP-NC, it is possible to extend the manufacturing lexicon to integrate the ongoing STEP-NC research. Figure 11.1 shows how the addition of further technologies to the manufacturing lexicon of the interoperable CAx framework, allows the findings of other researchers working with STEP-NC to be integrated within the framework.

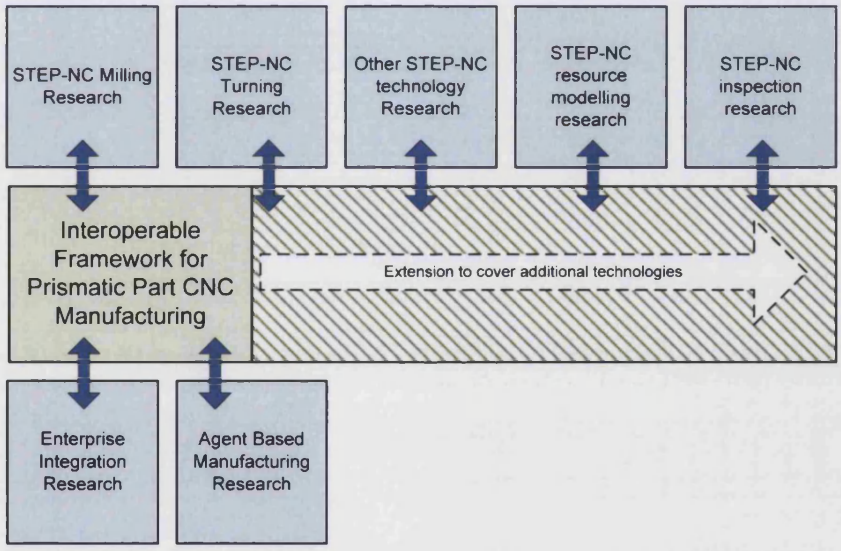


Figure 11.1 - Extension of the framework to integrate ongoing STEP-NC research

11.4.2. The Role of the Research in Realising the Manufuture Vision

The European Union initiative named Manufuture, provides a vision for the year 2020 with the strategic aim to "propose a strategy based on research and innovation, capable of speeding up the rate of industrial transformation in Europe, securing high added value employment and winning a major share of world manufacturing output in the future knowledge-driven

economy" (Manufuture, 2004). Figure 11.2 shows how the extension of the interoperable CAx framework, complies with and adds a new dimension to the Manufuture vision by considering the transition from isolated manufacturing resources of today to the interconnected knowledge oriented resources of the future.

The interoperable CAx framework with its extensible nature enables a smooth shift towards new markets, new products and new resources while maintaining the usability of the current resources for creating current products for current markets.

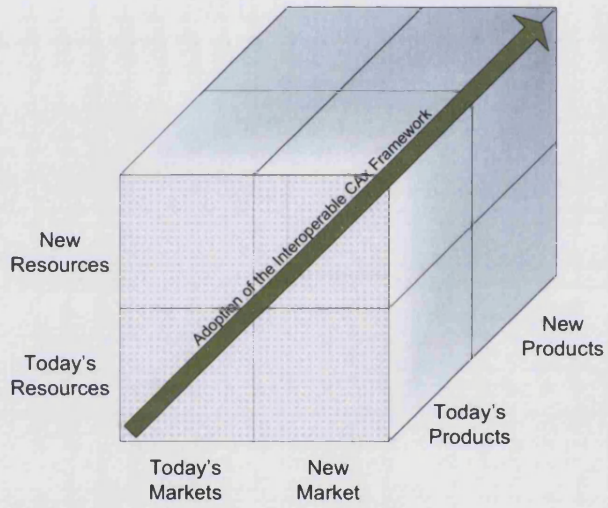


Figure 11.2 - Utilising the research to realise the Manufuture vision for interoperability

11.4.3. The Application of Service Oriented Architecture

Service orientation and Service Oriented Architecture (SOA) are relatively new technologies for integration. Microsoft defines service orientation as "a means for integrating across diverse systems. Each IT resource, whether an application, system, or trading partner, can be accessed as a service." (Microsoft, 2007)

Further development of the interoperable CAx platform could employ the various existing service oriented platforms. SOAP, UDDI and WSDL are technologies that should be investigated in future research on manufacturing interoperability in this context.

11.4.4. Utilisation of the Semantic Web

The semantic web is evolving as an extension of the world wide web that enables information to be stored and transferred in natural language as well as languages useful for software agents (w3c, 2007).

The facilities for ontology definition languages and semantic resource definitions together with a strong support for XML make the semantic web a viable technology for future developments of the Interoperable CAx framework.

11.4.5. Comprehensive Manufacturing Resource Models

An important and pivotal requirement for the wide adoption of the interoperable CAx framework is the availability of resource definitions for abstraction. The development of a comprehensive resource data model could enormously enhance the abstraction process by creating a generic definition method. CNC machine models and CAD/CAM system interface models, expressed in a common modelling language like UML would provide such a comprehensive resource model.

11.4.6. Towards a Universal Manufacturing Platform

The interoperable CAx framework, allows CAD/CAM/CNC resources to exchange information. By storing the data that is being exchanged cumulatively in a data warehouse, a comprehensive view of the manufacturing enterprise is created. A manufacturing knowledgebase to link various existing semantics in the data warehouse and a communication hub to physically transport data can be combined with this comprehensive data warehouse to create a universal manufacturing platform.

The universal manufacturing platform provides a robust, reliable and flexible foundation for CAx technology. Furthermore by employing additional technologies and models it is also possible to create interfaces between CAx system and business oriented information systems such as enterprise resource planning (ERP) or supply chain management (SCM) systems.

The realisation of the manufacturing platform could therefore take the "plug-and-manufacture" concept further to "plug, manufacture and verify" by incorporating computer-aided verification systems. Extending the context to business systems could empower the

platform to handle marketing, planning, procurement and distribution information as well as manufacturing data.

Figure 11.3 shows an overview of a proposition for the universal manufacturing platform. Various manufacturing information systems are connected to the communication hub via semantic adapters. The communication hub then passes the information to the enterprise knowledgebase where it is routed to the correct destination with a copy stored in the data warehouse.

To realise the platform, future research is required to specify and design a suitable enterprise data warehouse, develop a suitable knowledge engine to host the enterprise knowledgebase and to create an appropriate communication hub.

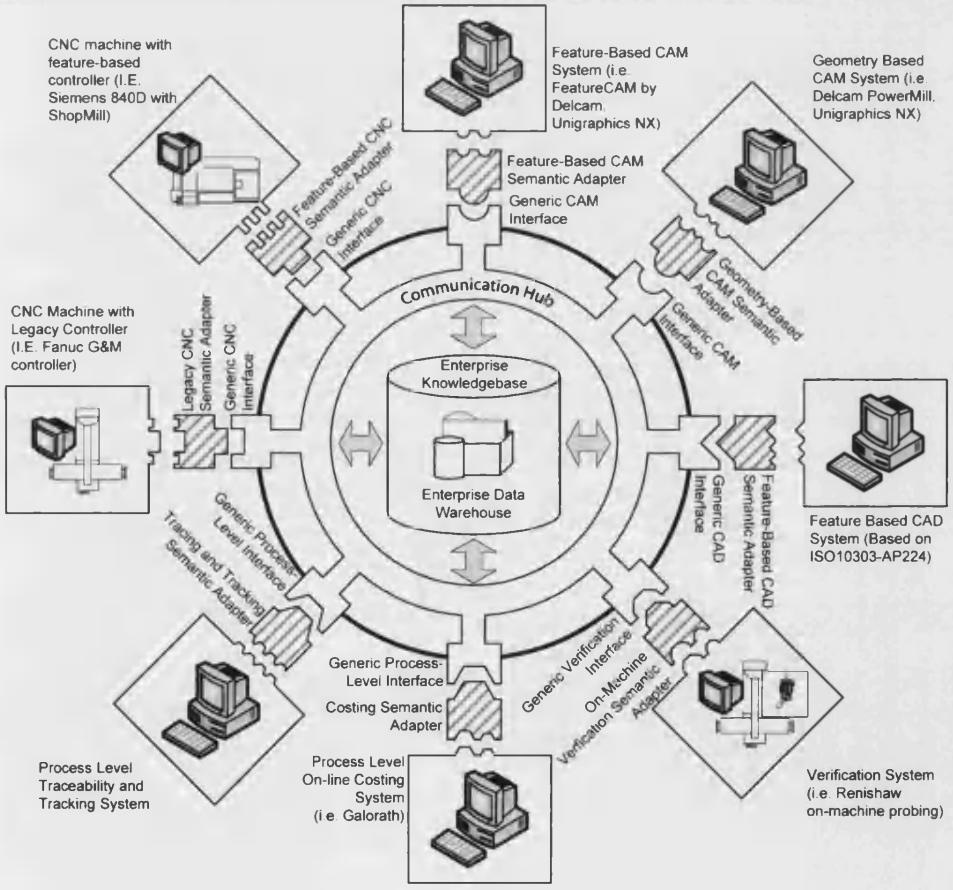


Figure 11.3 - An overview of the universal manufacturing platform

References

1. AFNOR (1989), Data exchange and transfer standard specification (SET), French Standard z68-300
2. Agapiou, J. (2006), Metal Cutting Theory and Practice (2nd Edition), CRC Press.
3. Ali, L.; Newman, S. & Petzing, J. (2005), 'Development of a STEP-compliant inspection framework for discrete components', *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture* **219**(7), 557-563.
4. Allen, R.; Harding, J. & Newman, S. (2005), 'The application of STEP-NC using agent-based process planning', *International Journal of Production Research* **43**(4), 655-670.
5. Alsene, E. (1999), 'The Computer Integration of the Enterprise', *IEEE Transactions on Engineering Management* **46**(1), 26-35.
6. Amaitik, S. & Engin Kilic, S. (2007), 'An intelligent process planning system for prismatic parts using STEP features', *International Journal of Advanced Manufacturing Technology* **31**, 978-993.
7. Autodesk (2007),'DXF Reference', Technical report, Autodesk Inc..
8. Bernard, A. & Perry, N. (2003), 'Fundamental concepts of product/technology/process informational integration for process modelling and process planning', *International Journal of Computer Integrated Manufacturing* **16**(7-8), 557-565.
9. Bernus, P.; Nemes, L. & Williams, T., ed. (1996), *Architectures for enterprise integration*, Kluwer Academic Publishers.
10. Bi, J.; Yu, T. & Li, Q. (2006), I2NC: A new type of computer numerical control, *in* 'Proceedings of the 2006 IEEE International Conference on Mechatronics and Automation', 803-808.
11. Booch, G (1994), Object-Oriented analysis and design with applications (2nd Edition), Addison-Wesley Professional.
12. Borselino, C.; Licari, R.; Lo Valvo, E. & Piacentin, M. (2004),An XML interface to STEP-NC systems for remote collaborative concurrent engineering, *in* 'Image Processing, Biomedicine, Multimedia, Financial Engineering and Manufacturing - Proceedings of the Sixth Biannual World Automation Congress', 413-418.
13. Brecher, C.; Vitr, M. & Wolf, J. (2006), 'Closed-loop CAPP/CAM/CNC process chain based on STEP and STEP-NC inspection tasks', *International Journal of Computer Integrated Manufacturing* **19**(6), 570-580.
14. Bryman, A. (2004), *Social Research Methods*, Oxford University Press, Oxford.
15. Bunse, C. & Gross, H. G. (2006), Unifying hardware and software components for embedded system development 'Architecting Systems with Trustworthy Components', Springer-Verlag Berlin, Berlin, 120-136.

16. Callen, J. (2005), 'STEP-NC is an advancement still searching for its markets', *Tooling and Production* **December 2005**, 21-23.

17. Chen, D. & Vernadat, F. (2004), 'Standards on enterprise integration and engineering - state of the art', *International Journal of Computer Integrated Manufacturing* **17**(3), 235-253.

18. Chen, X.; Zhang, C.; Lan, H.; Zhai, P. & Wu, H. (2005),A framework for CNC turning system based on STEP-NC, *in* 'Proceedings of SPIE - The International Society for Optical Engineering'.

19. Choi, I.; Suh, S.; Kim, K.; Song, M.; Jang, M. & Lee, B. (2006), 'Development process and data management of TurnSTEP: A STEP-compliant CNC system for turning', *International Journal of Computer Integrated Manufacturing* **19**(6), 546-558.

20. Clark, M.P. (2003), Data Networks, IP and the Internet: Networks, Protocols, Design and Operation, John Wiley and Sons

21. Da Silva, C. F.; Medini, L.; Ghafour, S. A.; Hoffmann, P.; Ghodous, P. & Lima, C. (2006), 'Semantic Interoperability of Heterogeneous Semantic Resources', *Electronic Notes in Theoretical Computer Science* **150**(2), 71-85.

22. Dan, B.; Li, L.; Zhang, X.; Guo, F. & Zhou, J. (2005), 'Network-integrated manufacturing system', *International Journal of Production Research* **43**(12), 2631-2647.

23. Dereli, T. & Baykasoglu, A. (2005), 'OPPS-PRI 2.0: An open and optimized process planning system for prismatic parts to improve the performance of SMEs in the machining industry', *International Journal of Production Research* **43**(5), 1039-1087.

24. DOD (1992),'MIL-PRF-28000A - Digital Representation for Communication of Product Data: IGES application subsets and IGES Application Protocols', Technical report, Deparment of Defence, US Military.

25. Dorf, R. & Kusiak, A. (1994), Handbook of Design, Manufacturing and Automation, Wiley-IEEE.

26. Doumeingts, G.; Vallespir, B. & Chen, D. (1995), 'Methodologies for designing CIM systems: A survey', *Computers in Industry* **25(3)**, 263-280.

27. Du, J.; Tian, X.; Zhang, Z.; Xu, J. & Zhu, M. (2005),Integrated CAD/CAM/CNC system based on STEP-NC and intelligent manufacturing, *in* 'Proceedings of SPIE - The International Society for Optical Engineering'

28. Eastman, C. & Augenbroe, F. (1998), Product Modelling Strategies for Today and the Future, *in* 'The Life-Cycle of IT innovations in Construction, Proc. CIB W78 conference (Bjork, B-C. & Jagbeck, A. eds.)', June 3-5, Stockholm, 191-207

29. Eastman, C. (1999), Building Product Models: Computer Environments Supporting Design and Construction, CRC Press.

30. ElMaraghy, H. (1993), 'Evolution and Future Perspectives of CAPP', *Annals of the CIRP* **42/2**, 739-751.

31. Euzenat, J. (2001),Towards a principled approach to semantic interoperability, *in* A. Gomez-Perez; M. Gruninger; H. Stuckenschmidt & M. Uschold, ed.,'Workshop on Ontologies and Information Sharing (IJCAI'01)'.

32. Fichtner, D.; Nestler, A.; Dang, T.; Schulze, A.; Carlsen, U.; Schreiber, S. & Lee, S. (2006), 'Use of agents and neural networks for acquisition and preparation of distributed NC information to support NC planning', *International Journal of Computer Integrated Manufacturing* **19**(6), 581-592.

33. Fortin, E.; Chatelain, J. & Rivest, L. (2004), 'An innovative software architecture to improve information flow from CAM to CNC', *Computers and Industrial Engineering* **46**(4), 655-667.

34. Gang, G.; Gui, G. & Wang, J. (2006), 'STEP-NC- and XML-enabled e-manufacturing', *International Journal of Computer Applications in Technology* **26**(1-2), 59-64.

35. Gao, J.; Aziz, H.; Maropoulos, P. & Cheung, W. (2003), 'Application of product data management technologies for enterprise integration', *International Journal of Computer Integrated Manufacturing* **16**(7-8), 491-500.

36. Garrido, J. (2003), Object-Oriented Programming: From Problem Solving to Java, Charles River Media.

37. Garrido Campos, J. & Hardwick, M. (2006), 'A traceability information model for CNC manufacturing', *CAD Computer Aided Design* **38**(5), 540-551.

38. Genesereth, M. & Fikes, R. (1992),'Knowledge Interchange Format, Version 3.0 Reference Manual'(Logic-92-1), Technical report, Computer Science Department, Stanford University.

39. Giachetti, R. (2004), 'A framework to review the information integration of the enterprise', *International Journal of Production Research* **42**(6), 1147-1166.

40. Goh, A.; Hui, S. C.; Song, B. & Wang, F. Y. (1994), 'A Study of SDAI implementation on object-oriented databases', *Computer Standards and Interfaces* **16,** 33-43

41. Goldfarb, C. & Prescod, P. (2004), *XML Handbook (5th Ed.)*, Prentice Hall PTR.

42. Grabowski, R. (2001), *CAD Manager's Guidebook*, Thomson Delmar Learning.

43. Hardwick, M.; Morris, K.; Spooner, D.; Rando, T. & Denno, P. (2000), 'Lessons learned developing protocols for the industrial virtual enterprise', *CAD Computer Aided Design* **32**(2), 159-166.

44. Hardwick, M. (2002),'Digital manufacturing using STEP-NC', *Technical Paper - Society of Manufacturing Engineers. MS* (MS02-252), Technical report, STEP Tools Inc.

45. Hardwick, M. (2002a), 'STEP into automatic machining', *Tooling and Production* **February 2002**, 38-39.

46. Hardwick, M. (2004), 'On STEP-NC and the complexities of product data integration', *Journal of Computing and Information Science in Engineering* **4**(1), 60-67.

47. Hardwick, M.; Kassel, B.; Crump, B. & Garrett, S. (2005), 'Improving shipyard manufacturing processes using STEP-NC', *Journal of Ship Production* **21**(3), 170-176.

48. Hardwick, M. & Loffredo, D. (2006), 'Lessons learned implementing STEP-NC AP-238', *International Journal of Computer Integrated Manufacturing* **19**(6), 523-532.

49. Heusinger, S.; Rosso Jr., R.; Klemm, P.; Newman, S. & Rahimifard, S. (2006), 'Integrating the CAx process chain for STEP-compliant NC manufacturing of asymmetric parts', *International Journal of Computer Integrated Manufacturing* **19**(6), 533-545.

50. Ho, K.; Newman, S. & Allen, R. (2005), 'STEP-NC compliant information modelling for wire electrical discharge machining component manufacture', *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture* **219**(10), 777-784.

51. Hoske, M. (2006), 'CNC Programming', *Control Engineering* **June 1,2006**.

52. IEC (1999),'IEC TC65/WG6 - Function Blocks for Industrial Process Measurement and Control Systems (Part-1: Architecture)', IEC.

53. ISO (1982),'ISO6983-1 - Numerical Control of Machines - Program Format and Definition of Address Words - Part 1: Data Format for Positioning, Line Motion and Contouring Control Systems.', International Standards Organization.

54. ISO (1994),'ISO10303-1 - Industrial Automation Systems and Integration - Product Data Representation and Exchange, Part 1. Overview and Fundamental Principles.', International Standards Organization.

55. ISO (1994a),'ISO10303-11 - Industrial automation systems and integration. Product data representation and exchange. Description methods: the EXPRESS language reference manual', International Standards Organisation.

56. ISO (1994b),'ISO10303-21 - Industrial automation systems and integration. Product data representation and exchange. Implementation methods. Clear text encoding of the exchange structure', International Standards Organisation.

57. ISO (1998),'ISO10303-22 - Industrial automation systems and integration -- Product data representation and exchange -- Part 22: Implementation methods: Standard data access interface', International Standards Organisation.

58. ISO (2000),'ISO10303-23 - Industrial automation systems and integration -- Product data representation and exchange -- Part 23: Implementation methods: C++ language binding to the standard data access interface', International Standards Organisation.

59. ISO (2000a),'ISO10303-27 - Industrial automation systems and integration -- Product data representation and exchange -- Part 27: Implementation methods: Java TM programming language binding to the standard data access interface with Internet/Intranet extensions', International Standards Organisation.

60. ISO (2001),'ISO10303-24 - Industrial automation systems and integration -- Product data representation and exchange -- Part 24: Implementation methods: C language binding of standard data access interface', International Standards Organisation.

61. ISO (2002),'ISO10303-204 - Industrial automation systems and integration -- Product data representation and exchange -- Part 204: Application protocol: Mechanical design using boundary representation', International Standards Organisation.

62. ISO (2003),'ISO10303-28 - Industrial automation systems and integration -- Product data representation and exchange -- Part 28: Implementation methods: XML representations of EXPRESS schemas and data', International Standards Organisation.

63. ISO (2003a),'ISO14649-1 - Industrial automation systems and integration - Physical Device Control - Data model for computerized numerical controllers - Part 1: Overview and Fundamental Principles.', International Standards Organization.

64. ISO (2003b),'ISO10303-214 - Industrial automation systems and integration -- Product data representation and exchange -- Part 214: Application protocol: Core data for automotive mechanical design processes', International Standards Organisation.

65. ISO (2004),'ISO14649-10 - Industrial automation systems and integration -- Physical device control -- Data model for computerized numerical controllers -- Part 10: General process data', International Standards Organisation.

66. ISO (2004a),'ISO14649-11 - Industrial automation systems and integration -- Physical device control -- Data model for computerized numerical controllers -- Part 11: Process data for milling', International Standards Organisation.

67. ISO (2004b),'ISO14649-111 - Industrial automation systems and integration -- Physical device control -- Data model for computerized numerical controllers -- Part 111: Tools for milling machines', International Standards Organisation.

68. ISO (2005),'ISO14649-12 - Industrial automation systems and integration -- Physical device control -- Data model for computerized numerical controllers -- Part 12: Process data for turning', International Standards Organisation.

69. ISO (2005a),'ISO14649-121 - Industrial automation systems and integration -- Physical device control -- Data model for computerized numerical controllers -- Part 121: Tools for turning machines', International Standards Organisation.

70. ISO (2005b),'ISO10303-203 - Industrial automation systems and integration -- Product data representation and exchange -- Part 203: Application protocol: Configuration controlled 3D design of mechanical parts and assemblies (modular version)', International Standards Organisation.

71. ISO (2006),'ISO10303-224 - Industrial automation systems and integration -- Product data representation and exchange -- Part 224: Application protocol: Mechanical product definition for process planning using machining features', International Standards Organisation.

72. ISO (2007),'ISO10303-238 - Industrial automation systems and integration -- Product data representation and exchange -- Part 238: Application protocol: Application interpreted model for computerized numerical controllers', International Standards Organisation.

73. Jennings, N. R. & Wooldridge, M.J. (1998) 'Applications of intelligent agents' in *Agent technology foundations, applications and markets,* Springer-Verlag.

74. Kochan, D. (1985), CAM : Developments in Computer-Integrated Manufacturing, Springer.

75. Kosanke, K. (1997), *Enterprise Engineering and Integration: Building International Consensus*, Springer-Verlag, chapter Enterprise integration and standardisation, 613-623.

76. Kosonen, M. & Doz, Y. (2006),'Fostering Strategic Agility: In Search for Renewed Growth', CKIR Workshop, August 29.

77. Kramer, T.; Proctor, F.; Xu, X. & Michaloski, J. (2006), 'Run-time interpretation of STEP-NC: Implementation and performance', *International Journal of Computer Integrated Manufacturing* **19**(6), 495-507.

78. Lee, W. & Bang, Y. (2003), 'Design and implementation of an ISO14649-compliant CNC milling machine', *International Journal of Production Research* **41**(13), 3007-3017.

79. Lee, W.; Bang, Y.; Ryou, M.; Kwon, W. & Jee, H. (2006), 'Development of a PC-based milling machine operated by STEP-NC in XML format', *International Journal of Computer Integrated Manufacturing* **19**(6), 593-602.

80. Leitao, P.; Colombo, A. & Restivo, F. (2005), 'ADACOR: A collaborative production automation and control architecture', *IEEE Intelligent Systems* **20**(1), 58-66.

81. Leitao, P. & Restivo, F. (2006), 'ADACOR: A holonic architecture for agile and adaptive manufacturing control', *Computers in Industry* **57**(2), 121-130.

82. Liang, Y.D. & Zhang H. (2007), *Computer Graphics Using Java 2D and 3D*, Prentice - Hall

83. Lin, H.; Harding, J. & Shahbaz, M. (2004), 'Manufacturing system engineering ontology for semantic interoperability across extended project teams', *International Journal of Production Research* **42**(24), 5099-5118.

84. Liu, R.; Zhang, C. & Newman, S. (2006), 'A framework and data processing for interfacing CNC with AP238', *International Journal of Computer Integrated Manufacturing* **19**(6), 516-522.

85. Liu, Y.; Guo, X.; Li, W.; K., Y.; Kashihara, K. & Fujishima, M. (2007), 'An intelligent NC program processor for CNC system of machine tool', *Robotics and Computer-Integrated Manufacturing* **23(2)**, 160-169.

86. Lubell, J. (2000),An XML Repository Architecture for STEP Modules, *in* 'Proceedings of the World Automation Congress, Maui, Hawaii, June 11-16'.

87. Lukka, K. (2003),The Constructive Research Approach, *in* L. Ojala & O.P. Hilmola, ed., 'Case Study Research in Logistics', Publications of the Turku School of Economics and Business Administration, , 83-101.

88. Lynch, M. (1992), 'Exploring distributed numerical control - using personal computers to store and retrieve numerical control programs and data', *Modern Machine Shop* **June 1992**.

89. Manufuture (2004), Manufuture: A vision for 2020, Assuring the Future of Manufacturing in Europe, Report of the High-Level Group

90. Mathews, C. (2002), 'Unlocking step-NC's potential', *Machine Design* **74**(20), 119-.

91. Mattson, M. (2001), *CNC Programming: Principles and Applications*, Thomson Delmar Learning.

92. Mehrabi, M. G.; Ulsoy, A. G. & Koren, Y. (2000), 'Reconfigurable manufacturing systems: Key to future manufacturing', *Journal of Intelligent Manufacturing* **11**(4), 403-419.

93. Mejabi, O. & Singh, N. (1997), 'A framework for enterprise-wide integration', *International Journal of Computer Integrated Manufacturing* **10**(1-4), 212-220.

94. Meyers, A. & Slattery, T. (2001), *Basic Machining Reference Handbook (2nd Edition)*, Industrial Press Inc..

95. Miao, H.; Sridharan, N. & Shah, J. (2002), 'CAD-CAM integration using machining features', *International Journal of Computer Integrated Manufacturing* **15**(4), 296-318.

96. Microsoft (2007) "Learn about service oriented architecture" website: http://www.microsoft.com/biztalk/solutions/soa/overview.mspx , accessed on 15 June 2007

97. Nacsa, J. (2001), Intelligent Open CNC System Based on the Knowledge Server Concept, *in* 'PROLAMAT', 360-368.

98. Newcomer, E. (2002), Understanding Web Services: XML, WSDL, SOAP and UDDI, Addison-Wesley Professional.

99. Newman, S.; Allen, R. & Rosso Jr., R. (2003), 'CAD/CAM solutions for STEP-compliant CNC manufacture', *International Journal of Computer Integrated Manufacturing* **16**(7-8), 590-597.

100. Newman, S. (2004), 'Learning a new language', *Machinery* **162**(4092), 10-14.

101. Ni, Q.; Yarlagadda, P. & Lu, W. (2006), 'Modeling of an integrated process planning system', *Computer-Aided Design and Applications* **3**(5), 567-576.

102. Ong, S. K.; Jiang, L. & Nee, A. Y. C. (2002), 'An Internet-Based Virtual CNC Milling System', *The International Journal of Advanced Manufacturing Technology* **20**(1), 20-30.

103. Ortiz, A.; Lario, F. & Ros, L. (1999), 'Enterprise Integration - Business Processes Integrated Management: A proposal for a methodology to develop Enterprise Integration Programs', *Computers in Industry* **40**(2), 155-171.

104. Parks, C. (1984), IGES as an interchange format for integrated circuit design, *in* '21st Conference on Design Automation', 273-274.

105. Parsons, J. & Stulen, F. (1958),'Motor controlled apparatus for positioning machine tools', United States Patent 2820187.

106. Peng, Y.; Finin, T.; Labrou, Y.; Cost, R.; Chu, B.; Long, J.; Tolone, W. & Boughannam, A. (1999), 'Agent-based approach for manufacturing integration: The ciimplex experience', *Applied Artificial Intelligence* **13**(1-2), 39-63.

107. Picco, G. (2001), 'Mobile Agents: an Introduction', *Microprocessors and Microsystems* **25**, 65-74.

108. Polywka, J. & Gabrel, S. (1992), *Programming of CNC Machines*, Industrial Press.

109. Pritschow, G.; Altintas, Y.; Jovane, F.; Koren, Y.; Mitsuishi, M.; Takata, S.; Van Brussel, H.; Weck, M. & Yamazaki, K. (2001), 'Open controller architecture - Past, present and future', *CIRP Annals - Manufacturing Technology* **50**(2), 463-470.

110. Proctor, F.; Michaloski, J. & Shackleford, W. (2002),Tying together design, process planning and machining with STEP-NC technology, *in* 'Proceedings of the 5th Biannual World Automation Congress', 33-38.

111. Radhakrishnan, S. S. & Raju, V. (2000), *CAD/CAM/CIM (2nd Edition)*, New Age International Publishers.

112. Rando, T. & McCabe, L. (1994), 'Issues in Implementing the C++ binding to SDAI', *Computer Standards and Interfaces* **16**, 331-340

113. Ray, S. & Jones, A. (2006), 'Manufacturing Interoperability', *Journal of Intelligent Manufacturing* **17**, 681-688.

114. Rosso, R. S. U.; Newman, S. T. & Rahimifard, S. (2004), 'The adoption of STEP-NC for the manufacture of asymmetric rotational components', *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture* **218**(11), 1639-1644.

115. Ryou, M.; Jee, H.; Kwon, W. & Bang, Y. (2006), 'Development of a data interface for rapid prototyping in STEP-NC', *International Journal of Computer Integrated Manufacturing* **19**(6), 614-626.

116. Saaski, J.; Salonen, T. & Paro, J. (2005), 'Integration of CAD, CAM and NC with STEP-NC', Technical report, VTT Industrial Systems.

117. Sandakly, F.; Garcia, J.; Ferreira, P. & Poyet, P. (2001), 'Distributed shared memory infrastructure for virtual enterprise in building and construction', *Journal of Intelligent Manufacturing* **12**, 199-212.

118. Schroeder, C. (1998), Printed Circuit Board Design Using AutoCAD, Newnes.

119. Schroeder, T. & Hoffmann, M. (2006), 'Flexible automatic converting of NC programs. A cross-compiler for structured text', *International Journal of Production Research* **44(13)**, 2671-2679.

120. Shen, W. & Norrie, D. (1999), 'Agent-Based Systems for Intelligent Manufacturing: A State-of-the-Art Survey', *Knowledge and Information Systems* **1**(2), 129-156.

121. Shen, W.; Lang, S. & Wang, L. (2005), 'iShopFloor: An internet-enabled agent-based intelligent shop floor', *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews* **35**(3), 371-381.

122. Shen, W.; Hao, Q.; Yoon, H. & Norrie, D. (2006), 'Applications of agent-based systems in intelligent manufacturing: An updated review', *Advanced Engineering Informatics* **20**(4), 415-431.

123. Shin, M. & Jung, M. (2004), 'MANPro: mobile agent-based negotiation process for distributed intelligent manufacturing', *International Journal of Production Research* **42**(2), 303-320.

124. Shin, S.; Suh, S. & Stroud, I. (2007), 'Reincarnation of G-code based part programs into STEP-NC for turning applications', *Computer-Aided Design* **39**, 1-16.

125. Smid, P. (2003), CNC Programming Handbook: A Comprehensive Guide to Practical CNC Programming, Industrial Press.

126. Smith, M. (2004), 'Portals: Toward an application framework for interoperability', *Communications of the ACM* **47**(10), 93-97.

127. Sokolov, A.; Richard, J.; Nguyen, V.; Stroud, I.; Maeder, W. & Xirouchakis, P. (2006), 'Algorithms and an extended STEP-NC-compliant data model for wire electro discharge machining based on 3D representations', *International Journal of Computer Integrated Manufacturing* **19**(6), 603-613.

128. Srinivasan, V. & Fischer, G. (1996), 'Direct interface integration of CAD and CAM software - A milling application', *Journal of Materials Processing Technology* **61**(1-2), 93-98.

129. Stevens, T. (2003), 'From Art to Part', *Industry Week* **December 2003**, 43.

130. Strasser, W. (1999), Theory and Practice of Geometric Modeling, Springer.

131. Stroud, I. & Xirouchakis, P. (2006), 'Strategy features for communicating aesthetic shapes for manufacturing', *International Journal of Computer Integrated Manufacturing* **19**(6), 639-649.

132. Suh, S.; Noh, S. & Choi, Y. (1995), 'A PC-based retrofitting toward CAD/CAM/CNC integration', *Computers and Industrial Engineering* **28**(1), 133-146.

133. Suh, S. & Cheon, S. (2002), 'A framework for an intelligent CNC and data model', *International Journal of Advanced Manufacturing Technology* **19**(10), 727-735.

134. Suh, S.; Cho, J. & Hong, H. (2002a), 'On the architecture of intelligent STEP-compliant CNC', *International Journal of Computer Integrated Manufacturing* **15**(2), 168-177.

135. Suh, S.; Chung, D.; Lee, B.; Cho, J.; Cheon, S.; Hong, H. & Lee, H. (2002b), 'Developing an integrated STEP-compliant CNC prototype', *Journal of Manufacturing Systems* **21**(5), 350-362.

136. Suh, S.; Lee, E.; Kim, H. & Cho, J. (2002c), 'Geometric error measurement of spiral bevel gears using a virtual gear model for STEP-NC', *International Journal of Machine Tools and Manufacture* **42**(3), 335-342.

137. Suh, S.; Lee, B.; Chung, D. & Cheon, S. (2003), 'Architecture and implementation of a shop-floor programming system for STEP-compliant CNC', *CAD Computer Aided Design* **35**(12), 1069-1083.

138. Suh, S.; Chung, D.; Lee, B.; Shin, S.; Choi, I. & Kim, K. (2006), 'STEP-compliant CNC system for turning: Data model, architecture, and implementation', *CAD Computer Aided Design* **38**(6), 677-688.

139. Sutherland, I. (1963), SKETCHPAD: A Man-Machine Graphical Communication System, *in* 'AFIPS, SJCC 23', 329-346.

140. Sutherland, I. (2003), 'SKETCHPAD: A man-machine graphical communication system' (574), Technical report, University of Cambridge Computer Laboratory.

141. Talavage, J. (1988), Flexible Manufacturing Systems in Practice, CRC Press.

142. Tolk, A. & Muguira, J. (2003), The levels of conceptual interoperability model (LCIM), *in* 'Proceedings of the IEEE Fall 2003 Simulation Interoperability Workshop', paper 03F-SIW-007.

143. Twigg, D.; Voss, C. & Winch, G. (1992), 'Implementing Integrating Technologies: Developing Managerial Integration for CAD/CAM', *International Journal of Operations and Production Management* **12**, 76-91.

144. Venkatesh, S.; Odendahi, D.; Xu, X.; Michaloski, J.; Proctor, F. & Kramer, T. (2005),Validating portability of STEP-NC tool center programming, *in* 'Proceedings of the ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference - DETC2005', 285-290.

145. Vetere, G. & Lenzerini, M. (2005), 'Models for semantic interoperability in service-oriented architectures', *IBM Systems Journal* **44**(4), 887-903.

146. Vinoski, S. (1997), 'CORBA: integrating diverse applications within distributed heterogeneous environments', *Communications Magazine, IEEE* **35**, 46-55.

147. W3C (2007), 'Semantic Web' on the Worldwide Web Consortium web page: http://www.w3.org/2001/sw/ , accessed on 15 June 2007

148. Wang, L.; Balasubramanian, S.; Norrie, D. & Brennan, R. (1998),Agent-based control system for next generation manufacturing, *in* 'IEEE International Symposium on Intelligent Control - Proceedings', 78-83.

149. Wang, L. (2002), DPP: A distributed process planning approach using function blocks, *in* 'Proceedings of the ASME Design Engineering Technical Conference', 387-394.

150. Wang, L.; Sams, R.; Verner, M. & Xi, F. (2003), 'Integrating Java 3D model and sensor data for remote monitoring and control', *Robotics and Computer-Integrated Manufacturing* **19**(1-2), 13-19.

151. Wang, L. & Shen, W. (2003), 'DPP: An agent-based approach for distributed process planning', *Journal of Intelligent Manufacturing* **14**(5), 429-439.

152. Wang, H. & Xu, X. (2004), A STEP-Compliant "Adaptor" for Linking CAPP with CNC, *in* 'Proceedings of the 34th International MATADOR Conference', 45-50.

153. Wang, L.; Orban, P.; Cunningham, A. & Lang, S. (2004), 'Remote real-time CNC machining for web-based manufacturing', *Robotics and Computer-Integrated Manufacturing* **20**(6 SPEC. ISS.), 563-571.

154. Wang, H.; Xu, X. & Des Tedford, J. (2006), 'Making a process plan adaptable to CNCs', *International Journal of Computer Applications in Technology* **26**(1-2), 49-58.

155. Wang, L.; Jin, W. & Feng, H. (2006a), 'Embedding machining features in function blocks for distributed process planning', *International Journal of Computer Integrated Manufacturing* **19**(5), 443-452.

156. Weck, M.; Wolf, J. & Kiritsis, D. (2001), STEP-NC The STEP Compliant NC Programming Interface: Evaluation and Improvement of the Modern Interface, *in* 'Proceedings of the IMS Project Forum'.

157. Werner Dankwort, C.; Weidlich, R.; Guenther, B. & Blaurock, J. (2004), 'Engineers' CAx education - it's not only CAD', *Computer-Aided Design* **36(14)**, 1439-1450.

158. Woolridge, M. (2002), An Introduction to Multiagent Systems, Wiley.

159. Wosnik, M.; Kramer, C.; Selig, A. & Klemm, P. (2006), 'Enabling feedback of process data by use of STEP-NC', *International Journal of Computer Integrated Manufacturing* **19**(6), 559-569.

160. Xu, X. & He, Q. (2002), Step-NC to re-shape manufacturing industry, *in* 'Proceedings of the 5th International Conference on Frontiers of Design and Manufacturing (ICFDM'2002)', 125-131.

161. Xu, X. & He, Q. (2004), 'Striving for a total integration of CAD, CAPP, CAM and CNC', *Robotics and Computer-Integrated Manufacturing* **20**(2), 101-109.

162. Xu, X. & Wang, J. (2004),Development of a G-Code free, STEP-compliant CNC lathe, *in* 'American Society of Mechanical Engineers, Computers and Information in Engineering Division, CED', 75-82.

163. Xu, X.; Wang, H.; Mao, J.; Newman, S.; Kramer, T.; Proctor, F. & Michaloski, J. (2005), 'STEP-compliant NC research: The search for intelligent CAD/CAPP/CAM/CNC integration', *International Journal of Production Research* **43**(17), 3703-3743.

164. Xu, X. (2006), 'Realization of STEP-NC enabled machining', *Robotics and Computer-Integrated Manufacturing* **22**(2), 144-153.

165. Xu, X. (2006a), *Database Modeling for Industrial Data Management: Emerging Technologies and Applications*, Idea Group Inc (IGI), chapter STEP-NC to Complete Product Development Chain, 148-184.

166. Xu, X. & Newman, S. (2006), 'Making CNC machine tools more open, interoperable and intelligent - A review of the technologies', *Computers in Industry* **57**(2), 141-152.

167. Xu, X.; Wang, L. & Rong, Y. (2006), 'STEP-NC and function blocks for interoperable manufacturing', *IEEE Transactions on Automation Science and Engineering* **3**(3), 297-307.

168. Zaidat, A.; Boucher, X. & Vincent, L. (2005), 'A framework for organization network engineering and integration', *Robotics and Computer-Integrated Manufacturing* **21**(3), 259-271.

169. Zarli, A. & Amar, V. (1997) Integrating STEP and CORBA for applications interoperability in future virtual enterprises computer-based infrastructures, *in* 'Proceedings of IASTED international conference on Intelligent Information Systems', 309

170. Zhang, Y.; Zhang, C. & Wang, H. P. (2000), 'An Internet Based STEP Data Exchange Framework for Virtual Enterprises', *Computers in Industry* **41**, 51-63.

171. Zhang, C.; Liu, R. & Hu, T. (2006), 'On the futuristic machine control in a STEP-compliant manufacturing scenario', *International Journal of Computer Integrated Manufacturing* **19**(6), 508-515.

172. Zhao, G.; Deng, J. & Shen, W. (2001), 'CLOVER: an agent-based approach to systems interoperability in cooperative design systems', *Computer in Industry* **45**, 261-276.

173. Zhao, F.; Xu, X. & Xie, S. (2007), 'STEP-NC enabled on-line inspection in support of closed-loop machining', *Robotics and Computer-Integrated Manufacturing* **In Press, Corrected Proof**.

174. Zhou, G.; Jiang, P. & Fukuda, S. (2002), 'Using mobile agents to schedule a manufacturing chain on the internet', *Concurrent Engineering: Research and Applications* **10**(4), 311-323.

175. Zhou, G. & Jiang, P. (2005), 'Using mobile agents to encapsulate manufacturing resources over the Internet', *The International Journal of Advanced Manufacturing Technology* **25**(1-2), 189-197.

176. Zhou, B.; Tang, J. & He, Z. (2005),An adaptive model of virtual enterprise based on dynamic web service composition, *in* 'Proceedings - Fifth International Conference on Computer and Information Technology, CIT 2005', 284-289.

177. Zimmermann, J.; Haasis, S. & van Houten, F. (2002), 'ULEO - Universal Linking of Engineering Objects', *Annals of the CIRP* **51/1**, 99-102.

Appendix A. Publications Based on the Research

A number of scientific peer reviewed papers were published by the author during the course of the research documented in this thesis. In this appendix, the titles and, in the case of journal papers, the abstracts for these papers are presented.

A.1. Journal Articles

1. **A. Nassehi**, S.T. Newman and R.D. Allen, 'The application of multi-agent systems for STEP-NC computer aided process planning of prismatic components'
International Journal of Machine Tools and Manufacture, Volume 46, Issue 5, April 2006, Pages 559-574

For many years, manufacturing firms have been seeking more efficient ways of manufacturing components with CNC machines. The emerging standards ISO 14649 and ISO 10303 (AP238) present an opportunity to revolutionize the way CNC machines are traditionally programmed. These standards better known as STEP-NC replace the traditional tool movement description languages with hierarchical data structures that allow a new breed of CNC to store part geometry together with the working steps of the operations required to manufacture the part.

STEP-NC provides the ability to store and utilise high level and detailed information from the CAD system to the intelligent STEP compliant CNC controller. With the advent of STEP-NC, computer aided process planning has become a critical link in the CAx process chain with the major requirement to generate interoperable process plans. The authors therefore believe it is necessary to redefine CAPP to reflect the change from the traditional tool movement based programming to STEP-NC based programming.

This paper examines the application of distributed artificial intelligence methods, namely collaborative multi-agent systems in designing an object-oriented process planning system for prismatic components in a STEP-NC compliant environment. The specification and design of a prototype system entitled the Multi-Agent System for Computer Aided Process Planning (MASCAPP) is outlined. Two test components have been designed, process planned, simulated on the machine controller and finally machined, to demonstrate the capabilities of the system and illustrate the activities required to implement STEP compliant manufacturing.

2. **A. Nassehi**, S.T. Newman and R.D. Allen, 'STEP-NC compliant process planning as an enabler for adaptive global manufacturing'
Robotics and Computer-Integrated Manufacturing, Volume 22, Issues 5-6, October-December 2006, Pages 456-467

Manufacturing firms are seeking more efficient methods of CNC manufacture. ISO14649 informally known as STEP-NC has been proposed as a high-level hierarchical manufacturing information model as a replacement for the low-level machining instructions of ISO6983 and RS274D. In this paper, the applicability of STEP-NC as an enabler for creating an adaptive global manufacturing system is examined. The overall framework of the system is presented followed by an outline of its information requirements. Suitability of STEP-NC to support each requirement is then studied with the necessary additions highlighted. Finally, a test component is used in conjunction with a prototype of the advanced global

manufacturing system to demonstrate the applicability of the STEP-NC standard to support manufacturing information in such a system.

3. **A. Nassehi**, R.D. Allen and S.T. Newman, 'Application of mobile agents in interoperable STEP-NC compliant manufacturing'
International Journal of Production Research, Volume 44, Issues 18-19, 2006, Pages 4159-4174

As the ratio of transport costs to total product cost increases, manufacturers try to fulfil the demands of each geographical region with products manufactured within the same region. Since the manufacturing resources available at each venue can be varied, it is necessary to adapt the manufacturing process plan for each production facility. Traditionally, this process is carried out manually by engineers utilizing computer-aided design (CAD) files. With the advances in artificial intelligence technology and emergence of integration standards such as STEP, it is now possible to automate the CAD/computer-aided manufacturing (CAM)/computer numerical control (CNC) interface. Distributed artificial intelligence techniques and mobile agents in particular can be used to transfer information throughout the manufacturing network and construct the distributed knowledge-base required for the intelligent interoperable integration of product data models and manufacturing resources. This paper explores the application of mobile agents as enablers of interoperability in a global manufacturing enterprise. A novel manufacturing chain based on the STEP-NC standard is proposed and the application of agents for transfer of manufacturing information is studied. The research is demonstrated through the use of a prototype manufacturing decision support system developed using agents.

4. S.T. Newman and **A. Nassehi**, 'Universal Manufacturing Platform for CNC Machining'
CIRP Annals - Manufacturing Technology, Volume 56, Issue 1, 2007, Pages 459-462

Today, CNC technology is a major contributor to the production capacity of industrial companies. The current NC standards only allow rudimentary low-bandwidth information transfer between various resources. A complex network of post-processors is therefore needed for the basic functionality of CAD/CAM/CNC systems. In this paper, the authors investigate and design a universal platform for supporting CNC manufacturing. The platform shifts the necessary knowledge transformations from the vendor specific software domain to the conceptual model space. This will eliminate the requirement for postprocessors. Consequently, resources will be interchangeable and interoperable, adding to the strategic agility of the manufacturing network.

5. Sanjeev Kumar, **Aydin Nassehi**, Stephen T. Newman, Richard D. Allen and Manoj K. Tiwari, 'Process control in CNC manufacturing for discrete components: A STEP-NC compliant framework'
*Robotics and Computer-Integrated Manufacturing, **In Press, Corrected Proof**, Available online 2 April 2007*

With today's highly competitive global manufacturing marketplace, the pressure for right-first-time manufacture has never been so high. New emerging data standards combined with machine data collection methods, such as in-process verification lead the way to a complete paradigm shift from the traditional manufacturing and inspection to intelligent networked process control. Low-level G and M codes offer very limited information on machine capabilities or work piece characteristics which consequently, results in no information being available on manufacturing processes, inspection plans and work piece attributes in terms of tolerances, etc. and design features to computer numerically controlled (CNC) machines. One solution to the aforementioned problems is using STEP-NC (ISO 14649) suite of standards, which aim to provide higher-level information for process control. In this paper, the authors provide a definition for

process control in CNC manufacturing and identify the challenges in achieving process control in current CNC manufacturing scenario. The paper then introduces a STEP-compliant framework that makes use of self-learning algorithms that enable the manufacturing system to learn from previous data and results in eliminating the errors and consistently producing quality products. The framework relies on knowledge discovery methods such as data mining encapsulated in a process analyser to derive rules for corrective measures to control the manufacturing process. The design for the knowledge-based process analyser and the various process control mechanisms conclude the paper.

6. **Aydin Nassehi,** Riliang Liu and Stephen T. Newman, 'A new software platform to support feature based process planning for interoperable STEP-NC manufacture' *International Journal for Computer Integrated Manufacturing, Accepted for publication in the special issue for the INCOM2006 conference*

CNC manufacturing has evolved with the use of faster, more precise and more capable CNC controllers and machine tools. The enhancements in machine tools however have not been integrated under a common platform to support CAD/CAM/CNC software interoperability and as a result a plethora of standards are being used for these systems. ISO10303 (STEP) and ISO14649 (STEP-NC) seek to eliminate the barriers in the exchange of information in the CNC manufacturing chain and enable interoperability throughout the manufacturing software domain. With the progress on standardization and implementation, computer systems in the manufacturing process chain require evolution to support the STEP-compliant planning and manufacture. This paper introduces a novel software platform entitled the Integrated Platform for Process Planning and Control (IP^3AC) to support STEP-NC compliant process planning (CAPP/CAM). A prototype process planning system (PPS) based on the platform is then presented as a sample application in the light of future interoperable planning and manufacture. The PPS has been developed with the application of a two-stage strategy for STEP-NC part program generation, namely general workplan generation and specific workplan generation. The work is verified through the use of case study components.

7. **A. Nassehi,** S.T. Newman, X.W. Xu and R.S.U. Rosso Jr., 'Toward Interoperable CNC Manufacturing' *International Journal for Computer Integrated Manufacturing, Selected for publication in the special issue for the DET2006 conference*

Most manufacturing enterprises now employ CNC technology in their production chain. Improving the performance and flexibility of the CAD/CAM/CNC chain can therefore, have a significant effect on competitiveness of such enterprises. Whereas hardware capabilities of these systems have increased proactively over the last few decades, the software components have been updated reactively to support the enhancements found on the newer generation of CNC machines. This passive approach has led to severe incompatibilities between the various CAD/CAM/CNC solutions. This paper presents the impediments and issues arising from such incompatibilities and proposes a new framework to overcome these barriers in achieving interoperability in the CAD/CAM/CNC chain. In the suggested framework different components of the CAD/CAM/CNC chain can exchange information with one another regardless of their native standards. The different elements comprising the framework are demonstrated by utilising a test component in a dynamic manufacturing scenario.

A.2. Conference Papers

1. **A Nassehi**, S T Newman & R D Allen
'Application of Mobile Agents in Interoperable STEP-NC Compliant Manufacturing',
Proceedings of the 18th International Conference on Production Research, Naples, Italy, July 2005, Paper Ref. CAD/CAT, pp1-6.

2. R Liu, S T Newman, R D Allen, **A Nassehi**, S Rahimifard
'The Design of a Holonic Workstation for STEP-NC Manufacturing ',
Proceedings of the 18th International Conference on Production Research, Naples, Italy, July 2005, Paper Ref. Special Machining Processes

3. **A Nassehi**, S T Newman & R D Allen
'The application of multi-agent systems in STEP-NC compliant process planning of prismatic components',
Proceedings of the 15th International Conference on Flexible Automation and Intelligent Manufacturing Conference (FAIM2005), Bilbao, Spain, July 2005, pp.202-209.

4. **A Nassehi**, Richard D Allen, and S T Newman
'Intelligent Replication of Manufacturing Information between CAD/CAM Systems and CNC Controllers',
Proceedings of the 16th International Conference on Flexible Automation and Intelligent Manufacturing Conference (FAIM2006) Limerick, Ireland, June 2006, pp413-420.

5. S Kumar, **A Nassehi**, R D Allen, S T Newman, and M K Tiwari
'A STEP-Compliant Knowledge Based Systems for the process control of discrete components'
Proceedings of the 16th International Conference on Flexible Automation and Intelligent Manufacturing Conference (FAIM2006) Limerick, Ireland, June 2006, pp831-839.

6. R Liu, S T. Newman, R D Allen, **A Nassehi**
'Feature-Based Process Planning For Interoperable STEP-NC Manufacture',
Proceedings of the 12th IFAC Symposium on Information Control Problems in Manufacturing (INCOM 2006), Saint-Etienne, France, 17-19 May 2006.

7. **A Nassehi**, R D Allen & S T Newman
'A New Software Platform for STEP-NC Manufacturing Application Development ',
Proceedings of the 12th IFAC Symposium on Information Control Problems in Manufacturing (INCOM 2006), Saint-Etienne, France, 17-19 May 2006.

8. **A Nassehi**, S T Newman, X W Xu, R D. Allen & R S U Rosso Jr.
'Adaptability and Interoperability in CNC Manufacturing',
Proceedings of the 3rd CIRP international conference on Digital Enterprise Technology' (DET06), Setubal, Portugal, 18th -20th September 2006.

9. M Bachlaus, M K Tiwari, S Kumar, **A Nassehi**, S T Newman
'Web Based Multi Agent Platform for Collaborative Manufacturing',
Proceedings of the 3rd CIRP international conference on Digital Enterprise Technology' (DET06), Setubal, Portugal, 18th -20th September 2006

10. R L Liu, C R Zhang, **A Nassehi** and S T Newman
'A STEP-NC programming system for prismatic parts',
Proceedings of the 12th International Manufacturing Conference in China (IMCC'2006), Xi'an, China, September 2006.

11. L Zheng, H Dong, P Vichare, **A Nassehi**, S T Newman and J Guo
'A Hierarchical Case Process Knowledge Model for Supporting Rapid Process Configuration',
Accepted for presentation in the 17th International Conference on Flexible Automation and Intelligent Manufacturing Conference (FAIM2007) Philadelphia, USA, June 2007

12. S T Newman, **A Nassehi**, X W Xu, R S U Rosso, L Wang, Y Yusof, L Ali, R Liu, L Zheng, S Kumar, P Vichare, V Dhokia
'Interoperable CNC for Global Manufacturing'
Accepted Keynote presentation in the 17th International Conference on Flexible Automation and Intelligent Manufacturing Conference (FAIM2007) Philadelphia, USA, June 2007

13. S T Newman, L Ali, A Brail, C Brecher, P Klemm, R Liu, **A Nassehi**, V K Nguyen, F Proctor, R S U Rosso, I Stroud, S-H. Suh, M Vitr, L Wang and X W Xu
'The Evolution of CNC Technology from Automated Manufacture to Global Interoperable Manufacturing'
Accepted keynote presentation in the 2nd International Conference on Changeable, Agile, Reconfigurable and Virtual Production (CARV2007), Toronto, Canada, July 2007

14. P Vichare, **A Nassehi**, S Kumar, S Newman, L Zheng, V Dhokia
'Towards a STEP-NC compliant resource model for machine tools'
Submitted for presentation in the 4th CIRP international conference on Digital Enterprise Technology (DET2007), Bath, UK, September 2007

Appendix B. Program Listings for the Test Component

B.1. ShopMill MPF Listing

```
E_HEAD(3674111,0.,0.,0.,150.,100.,-50.,71,17,10.,100.,1,0,6,);*RO*
N5 E_MI_PL("63mmFaceMill","16mmSlotDrill",1,2500.,1,4000.,1,9,0.,90,-5.,90,2.5,0.3,-70.,90,-
70.,90,220.,90,170.,90,60.,1);*RO*
E_CON("OUT1",1,"E_LAB_A_OUT1","E_LAB_E_OUT1");*RO*
E_CON("IN",1,"E_LAB_A_IN","E_LAB_E_IN");*RO*
N10 E_CP_CO("16mmSlotDrill","8mmDrill",1,2500.,1,4000.,1,"2007052211123300",1,8.,-4.5,-
30.,90,20.,3.,0.5,0.5,0.,0.,1000.,1,0.,100.,10.,1,"0000471611117216893","0000059009188055010
",0,1,1.,1.,"",1,27.008);*RO*
N90 E_CP_CO("16mmSlotDrill","8mmDrill",1,2500.,1,4000.,1,"2007052211123301",2,8.,-4.5,-
30.,90,20.,3.,0.5,0.5,0.,0.,1000.,1,0.,100.,10.,1,"0000450850936900260 5","0000005870346461130
8",0,1,1.,1.,"",1,27.008);*RO*
E_CON("CLEAR_OUT",1,"E_LAB_A_CLEAR_OUT","E_LAB_E_CLEAR_OUT");*RO*
E_CON("FIST_OUT",1,"E_LAB_A_FIST_OUT","E_LAB_E_FIST_OUT");*RO*
N35 E_CP_CO("16mmSlotDrill","8mmDrill",1,2500.,1,4000.,1,"2007052211123402",1,8.,-4.5,-
15.,90,20.,3.,0.5,0.5,0.,0.,1000.,1,0.,100.,10.,1,"0000557451087596825 6","0000006180237710803 9
",0,1,1.,1.,"",1,33.344);*RO*
N95 E_CP_CO("16mmSlotDrill","8mmDrill",1,2500.,1,4000.,1,"2007052211123403",2,8.,-4.5,-
15.,90,20.,3.,0.5,0.5,0.,0.,1000.,1,0.,100.,10.,1,"0000532850669733247 9","0000006140665366433 7
",0,1,1.,1.,"",1,33.344);*RO*
E_CON("FIST_IN",1,"E_LAB_A_FIST_IN","E_LAB_E_FIST_IN");*RO*
N40 E_CP_CO("16mmSlotDrill","8mmDrill",1,2500.,1,4000.,1,"2007052211123504",1,8.,-4.5,-
10.,90,20.,3.,0.5,0.5,0.,0.,1000.,1,0.,100.,10.,1,"0000110020935143871 4","0000006270344012571 5
",0,1,1.,1.,"",1,23.84);*RO*
N100 E_CP_CO("16mmSlotDrill","8mmDrill",1,2500.,1,4000.,1,"2007052211123505",2,8.,-4.5,-
10.,90,20.,3.,0.5,0.5,0.,0.,1000.,1,0.,100.,10.,1,"0000105320897171984 6","0000006230771668201 3
",0,1,1.,1.,"",1,23.84);*RO*
E_CON("PRIME",1,"E_LAB_A_PRIME","E_LAB_E_PRIME");*RO*
N45 E_CP_CO("16mmSlotDrill","8mmDrill",1,2500.,1,4000.,1,"2007052211123606",1,8.,-9.5,-
15.,90,20.,3.,0.5,0.5,0.,0.,1000.,1,0.,100.,10.,1,"0000108870746764271 9","0000006090541833166 3
",0,1,1.,1.,"",1,14.336);*RO*
N105 E_CP_CO("16mmSlotDrill","8mmDrill",1,2500.,1,4000.,1,"2007052211123607",2,8.,-9.5,-
15.,90,20.,3.,0.5,0.5,0.,0.,1000.,1,0.,100.,10.,1,"0000104220060285673 5","0000006050969488796 0
",0,1,1.,1.,"",1,14.336);*RO*
E_CON("SECOND",1,"E_LAB_A_SECOND","E_LAB_E_SECOND");*RO*
N50 E_CP_CO("16mmSlotDrill","8mmDrill",1,2500.,1,4000.,1,"2007052211123608",1,8.,-9.5,-
15.,90,20.,3.,0.5,0.5,0.,0.,1000.,1,0.,100.,10.,1,"0000108340748515084 9","0000006090541833166 3
",0,1,1.,1.,"",1,11.168);*RO*
N110 E_CP_CO("16mmSlotDrill","8mmDrill",1,2500.,1,4000.,1,"2007052211123709",2,8.,-9.5,-
15.,90,20.,3.,0.5,0.5,0.,0.,1000.,1,0.,100.,10.,1,"0000103710450673156 8","0000006050969488796 0
",0,1,1.,1.,"",1,11.168);*RO*
N55
E_PO_CIR(1,0,0,"16mmSlotDrill","8mmDrill",1,2500.,1,4000.,1,1,1000.,1,0.,0.,90,5.,91,3.,0.,90,
0.,90,20.,0.1,0.1,9.,1.,40.,1,1);*RO*
N115
E_PO_CIR(1,0,0,"16mmSlotDrill","8mmDrill",1,2500.,1,4000.,1,2,1000.,1,0.,0.,90,5.,91,3.,0.,90,
0.,90,20.,0.1,0.1,9.,1.,40.,-1,1);*RO*
N60 _E_P001: E_PS_SEQ(1,0,0,-
15.,90,87.5,90,80.,90,112.5,90,80.,90,75.,90,54.869,90,100.,90,54.869,90,125.,90,54.869,90,0.,
0,0.,0,0.,0,0.,0,0.,0,0.,0,0.,0,0.,0,1);*RO*
N65 E_PO_REC(4,0,0,"16mmSlotDrill","8mmDrill",1,2500.,1,4000.,1,1,1000.,1,0.,-
15.,90,5.,91,2.,1,17.5,90,50.,90,25.,40.,8.,0.,0.1,0.1,8.,3.,1.,60.,0,1);*RO*
N120 E_PO_REC(4,0,0,"16mmSlotDrill","8mmDrill",1,2500.,1,4000.,1,2,1000.,1,2.,-
14.5,90,5.,91,2.,1,17.5,90,50.,90,25.,40.,8.,0.,0.1,0.1,8.,3.,1.,60.,0,1);*RO*
N70 E_DR(1,0,0,"8mmDrill","3mmDrill",1,1000.,1,4000.,1,-40.,90,0.,1);*RO*
N75 _E_P002: E_PS_SEQ(1,0,0,-
9.5,90,106.,90,21.,90,11.,90,51.,90,50.,90,86.,90,87.5,90,80.,90,112.5,90,80.,90,136.,90,86.,9
0,0.,0,0.,0,0.,0,0.,0,0.,0,0.,0,0.,0,2);*RO*
N80 E_DR(1,0,0,"3mmDrill","",1,1000.,1,4000.,1,-40.,90,0.,1);*RO*
N85 _E_P003: E_PS_ROW(1,0,0,-15.,90,46.5,90,57.5,90,90.,5.,5,3);*RO*
M30 ;#SM;*RO*
```

```
E_LAB_A_OUT1: ;#SM Z:1
G17 G90 G71 DIAMOF
G0 X=-13 Y=103
G2 X=-3 Y=113 I=AC(-3) J=AC(103)
G1 X=153
G2 X=163 Y=103 I=AC(153) J=AC(103)
G1 Y=-3
G2 X=153 Y=-13 I=AC(153) J=AC(-3)
G1 X=-3
G2 X=-13 Y=-3 I=AC(-3) J=AC(-3)
G1 Y=103
E_LAB_E_OUT1:

E_LAB_A_IN: ;#SM Z:1
G17 G90 G71 DIAMOF
G0 X=5 Y=95
G1 X=145
G1 Y=5
G1 X=73
G2 X=55 Y=23 I=AC(73) J=AC(23)
G3 X=37 Y=45 I=AC(32.556) J=AC(23)
G1 X=11
G2 X=5 Y=51 I=AC(11) J=AC(51)
G1 Y=95
E_LAB_E_IN:



E_LAB_A_FIST_OUT: ;#SM Z:1
G17 G90 G71 DIAMOF
G0 X=60 Y=55
G2 X=81.708 Y=68.416 I=AC(75) J=AC(55)
G3 X=93.292 I=AC(87.5) J=AC(80)
G2 X=106.708 I=AC(100) J=AC(55)
G3 X=118.292 I=AC(112.5) J=AC(80)
G2 X=140 Y=55 I=AC(125) J=AC(55)
G1 Y=23
G2 X=127 Y=10 I=AC(127) J=AC(23)
G1 X=73
G2 X=60 Y=23 I=AC(73) J=AC(23)
G1 Y=55
E_LAB_E_FIST_OUT:

E_LAB_A_FIST_IN: ;#SM Z:1
G17 G90 G71 DIAMOF
G0 X=65 Y=55
G2 X=79.474 Y=63.943 I=AC(75.002) J=AC(54.997)
G3 X=95.526 I=AC(87.5) J=AC(80)
G2 X=104.474 I=AC(100) J=AC(54.993)
G3 X=120.526 I=AC(112.5) J=AC(79.988)
G2 X=135 Y=55 I=AC(125) J=AC(55)
G1 Y=23
G2 X=127 Y=15 I=AC(127) J=AC(23)
G1 X=118
G1 X=94
G1 X=73
G2 X=65 Y=23 I=AC(73) J=AC(23)
G1 Y=35
G1 Y=47
G1 Y=55
E_LAB_E_FIST_IN:

E_LAB_A_PRIME: ;#SM Z:1
G17 G90 G71 DIAMOF
G0 X=100 Y=42
G1 X=70
G2 X=65 Y=47 I=AC(70) J=AC(47)
G1 Y=55
G2 X=79.474 Y=63.943 I=AC(75.002) J=AC(54.997)
```

```
G3 X=95.526 I=AC(87.5) J=AC(80)
G2 X=104.474 I=AC(100) J=AC(54.993)
G3 X=120.526 I=AC(112.5) J=AC(79.988)
G2 X=135 Y=55 I=AC(125) J=AC(55)
G1 Y=23
G2 X=127 Y=15 I=AC(127) J=AC(23)
G1 X=118
G2 X=112 Y=21 I=AC(118) J=AC(21)
G3 X=109.5 Y=25.873 I=AC(106) J=AC(21)
G2 X=107 Y=30.747 I=AC(113) J=AC(30.747)
G1 Y=35
G3 X=100 Y=42 I=AC(100) J=AC(35)
E_LAB_E_PRIME:

E_LAB_A_SECOND: ;#SM Z:1
G17 G90 G71 DIAMOF
G0 X=70 Y=40
G1 X=100
G2 X=105 Y=35 I=AC(100) J=AC(35)
G1 Y=28.746
G2 X=103.529 Y=26.468 I=AC(102.5) J=AC(28.746)
G3 X=100 Y=21 I=AC(106) J=AC(21)
G2 X=94 Y=15 I=AC(94) J=AC(21)
G1 X=73
G2 X=65 Y=23 I=AC(73) J=AC(23)
G1 Y=35
G2 X=70 Y=40 I=AC(70) J=AC(35)
E_LAB_E_SECOND:
E_LAB_A_CLEAR_OUT: ;#SM Z:2
G17 G90 DIAMOF ;*GP*
G0 X0 Y108 ;*GP*
G1 X150 ;*GP*
G2 X158 Y100 I=AC(150) J=AC(100) ;*GP*
G1 Y0 ;*GP*
G2 X150 Y-8 I=AC(150) J=AC(0) ;*GP*
G1 X73 ;*GP*
G2 X42 Y23 I=AC(73) J=AC(23) ;*GP*
G3 X35.419 Y32 I=AC(32.556) J=AC(23) ;*GP*
G1 X0 ;*GP*
G2 X-8 Y40 I=AC(0) J=AC(40) ;*GP*
G1 Y100 ;*GP*
G2 X0 Y108 I=AC(0) J=AC(100) ;*GP*
RET ;*GP*
E_LAB_E_CLEAR_OUT:
```

B.2. STEP-NC listing

```
ISO-10303-21;
HEADER;
FILE_DESCRIPTION(('ISO 14649-11 FILE','AUTOMATIC OUTPUT GENERATED BY INTEROPERABLE
FRAMEWORK'),'1');
FILE_NAME('EXAMPLE.STP','2007-05-07',('AYDIN NASSEHI','STEPHEN NEWMAN'),('BATH
UNIVERSITY'),$,'ISO 14649',$);
FILE_SCHEMA(('combined_schema'));
ENDSEC;
DATA;
#1=PROJECT('Aerospace Test',#2,(#3),$,$,$);
#2=WORKPLAN('Main
Workplan',(#4,#5,#6,#7,#8,#9,#10,#11,#12,#13,#25,#26,#27,#28,#29,#30,#31,#32,#34,#35,#36,#37,#
14,#15,#16,#17,#18,#19,#20,#21,#22,#23,#24,#33),$,$,$);
#3=WORKPIECE('Simple Workpiece',#53,0.01,$,$,#54,());
#4=MACHINING_WORKINGSTEP('Milling working step for roughing pocket',#38,#39,#40,$);
#5=MACHINING_WORKINGSTEP('Milling working step for finishing pocket',#38,#39,#45,$);
#6=MACHINING_WORKINGSTEP('Milling working step for roughing pocket',#176,#177,#178,$);
#7=MACHINING_WORKINGSTEP('Milling working step for finishing pocket',#176,#177,#183,$);
#8=MACHINING_WORKINGSTEP('Milling working step for roughing pocket',#345,#346,#347,$);
#9=MACHINING_WORKINGSTEP('Milling working step for finishing pocket',#345,#346,#352,$);
#10=MACHINING_WORKINGSTEP('Milling working step for roughing pocket',#450,#451,#452,$);
#11=MACHINING_WORKINGSTEP('Milling working step for finishing pocket',#450,#451,#457,$);
#12=MACHINING_WORKINGSTEP('Milling working step for roughing pocket',#575,#576,#577,$);
#13=MACHINING_WORKINGSTEP('Milling working step for finishing pocket',#575,#576,#582,$);
#14=MACHINING_WORKINGSTEP('Drilling',#661,#662,#663,$);
#15=MACHINING_WORKINGSTEP('Drilling',#679,#680,#681,$);
#16=MACHINING_WORKINGSTEP('Drilling',#697,#698,#699,$);
#17=MACHINING_WORKINGSTEP('Drilling',#715,#716,#717,$);
#18=MACHINING_WORKINGSTEP('Drilling',#733,#734,#735,$);
#19=MACHINING_WORKINGSTEP('Drilling',#751,#752,#753,$);
#20=MACHINING_WORKINGSTEP('Drilling',#769,#770,#771,$);
#21=MACHINING_WORKINGSTEP('Drilling',#787,#788,#789,$);
#22=MACHINING_WORKINGSTEP('Drilling',#805,#806,#807,$);
#23=MACHINING_WORKINGSTEP('Drilling',#823,#824,#825,$);
#24=MACHINING_WORKINGSTEP('Drilling',#841,#842,#843,$);
#25=MACHINING_WORKINGSTEP('Milling working step for roughing pocket',#859,#860,#861,$);
#26=MACHINING_WORKINGSTEP('Milling working step for finishing pocket',#859,#860,#866,$);
#27=MACHINING_WORKINGSTEP('Milling working step for roughing pocket',#890,#891,#892,$);
#28=MACHINING_WORKINGSTEP('Milling working step for finishing pocket',#890,#891,#897,$);
#29=MACHINING_WORKINGSTEP('Milling working step for roughing pocket',#914,#915,#916,$);
#30=MACHINING_WORKINGSTEP('Milling working step for finishing pocket',#914,#915,#921,$);
#31=MACHINING_WORKINGSTEP('Milling working step for roughing pocket',#938,#939,#940,$);
#32=MACHINING_WORKINGSTEP('Milling working step for finishing pocket',#938,#939,#945,$);
#33=MACHINING_WORKINGSTEP('Drilling',#962,#963,#964,$);
#34=MACHINING_WORKINGSTEP('Milling working step for roughing pocket',#980,#981,#982,$);
#35=MACHINING_WORKINGSTEP('Milling working step for finishing pocket',#980,#981,#987,$);
#36=MACHINING_WORKINGSTEP('Milling working step for roughing pocket',#1004,#1005,#1006,$);
#37=MACHINING_WORKINGSTEP('Milling working step for finishing pocket',#1004,#1005,#1011,$);
#38=PLANE('Security plane for pocket',#41);
#39=CLOSED_POCKET('pocket',#3,(#40,#45),#46,#47,(#48),$,#49,#50,#51,#52);
#40=BOTTOM_AND_SIDE_ROUGH_MILLING($,$,'Rough milling for pocket',$,$,$,$,$,$,$,$,$,$,$,$);
#41=AXIS2_PLACEMENT_3D('Axis for Security axis placement',#42,#43,#44);
#42=CARTESIAN_POINT('point for Security axis placement',(0.0,0.0,10.0));
#43=DIRECTION('axis direction for Security axis placement',(0.0,0.0,1.0));
#44=DIRECTION('reference direction for Security axis placement',(1.0,0.0,0.0));
#45=BOTTOM_AND_SIDE_FINISH_MILLING($,$,'Rough milling for pocket',$,$,$,$,$,$,$,$,$,$,$,$);
#46=AXIS2_PLACEMENT_3D('Axis for pocket placement',#60,#61,#62);
#47=PLANE('Pocket Depth plane',#63);
#48=BOSS('pocket',#3,(#40,#45),#67,#68,#69,0.0);
#49=PLANAR_POCKET_BOTTOM_CONDITION();
#50=TOLERANCED_LENGTH_MEASURE(2.0,$);
#51=TOLERANCED_LENGTH_MEASURE(0.05,$);
#52=GENERAL_CLOSED_PROFILE(#125,#126);
#53=MATERIAL('Test Material','TST-1',(#55));
#54=BLOCK('block for Simple Workpiece',#56,150.0,100.0,50.0);
#55=NUMERIC_PARAMETER('E',2100000.0,'Kg/m4');
#56=AXIS2_PLACEMENT_3D('Axis for axis for block for Simple Workpiece',#57,#58,#59);
```

```
#57=CARTESIAN_POINT('point for axis for block for Simple Workpiece',(0.0,0.0,-50.0));
#58=DIRECTION('axis direction for axis for block for Simple Workpiece',(0.0,0.0,1.0));
#59=DIRECTION('reference direction for axis for block for Simple Workpiece',(1.0,0.0,0.0));
#60=CARTESIAN_POINT('point for pocket placement',(0.0,0.0,0.0));
#61=DIRECTION('axis direction for pocket placement',(0.0,0.0,1.0));
#62=DIRECTION('reference direction for pocket placement',(1.0,0.0,0.0));
#63=AXIS2_PLACEMENT_3D('Axis for Depth placement',#64,#65,#66);
#64=CARTESIAN_POINT('point for Depth placement',(0.0,0.0,-30.0));
#65=DIRECTION('axis direction for Depth placement',(0.0,0.0,1.0));
#66=DIRECTION('reference direction for Depth placement',(1.0,0.0,0.0));
#67=AXIS2_PLACEMENT_3D('Axis for pocket placement',#70,#71,#72);
#68=PLANE('Pocket Depth plane',#73);
#69=GENERAL_CLOSED_PROFILE(#77,#78);
#70=CARTESIAN_POINT('point for pocket placement',(0.0,0.0,0.0));
#71=DIRECTION('axis direction for pocket placement',(0.0,0.0,1.0));
#72=DIRECTION('reference direction for pocket placement',(1.0,0.0,0.0));
#73=AXIS2_PLACEMENT_3D('Axis for Depth placement',#74,#75,#76);
#74=CARTESIAN_POINT('point for Depth placement',(0.0,0.0,-30.0));
#75=DIRECTION('axis direction for Depth placement',(0.0,0.0,1.0));
#76=DIRECTION('reference direction for Depth placement',(1.0,0.0,0.0));
#77=AXIS2_PLACEMENT_3D('Axis for profile placement',#79,#80,#81);
#78=COMPOSITE_CURVE('E_LAB_A_IN Curve',(#82,#83,#84,#85,#86,#87,#88,#89),.F.);
#79=CARTESIAN_POINT('point for profile placement',(0.0,0.0,0.0));
#80=DIRECTION('axis direction for profile placement',(0.0,0.0,1.0));
#81=DIRECTION('reference direction for profile placement',(1.0,0.0,0.0));
#82=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#90);
#83=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#93);
#84=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#96);
#85=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#99);
#86=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#107);
#87=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#113);
#88=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#116);
#89=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#122);
#90=POLYLINE('Polyline for contour ',(#91,#92));
#91=CARTESIAN_POINT('Poly point 1',(5.0,95.0,0.0));
#92=CARTESIAN_POINT('Poly point 2',(145.0,95.0,0.0));
#93=POLYLINE('Polyline for contour ',(#94,#95));
#94=CARTESIAN_POINT('Poly point 1',(145.0,95.0,0.0));
#95=CARTESIAN_POINT('Poly point 2',(145.0,5.0,0.0));
#96=POLYLINE('Polyline for contour ',(#97,#98));
#97=CARTESIAN_POINT('Poly point 1',(145.0,5.0,0.0));
#98=CARTESIAN_POINT('Poly point 2',(73.0,5.0,0.0));
#99=TRIMMED_CURVE('Trimmed Curve',#100,(#101),(#102),.T.,.CARTESIAN.);
#100=CIRCLE('Circle1',#103,18.0);
#101=CARTESIAN_POINT('Trim point 1',(73.0,5.0,0.0));
#102=CARTESIAN_POINT('Trim point 2',(55.0,23.0,0.0));
#103=AXIS2_PLACEMENT_3D('Circle Placement',#104,#105,#106);
#104=CARTESIAN_POINT('Circle centre',(73.0,23.0,0.0));
#105=DIRECTION('Z Direction',(0.0,0.0,1.0));
#106=DIRECTION('X Direction',(1.0,0.0,0.0));
#107=TRIMMED_CURVE('Trimmed Curve',#108,(#109),(#110),.T.,.CARTESIAN.);
#108=CIRCLE('Circle1',#111,22.444000000000003);
#109=CARTESIAN_POINT('Trim point 2',(37.0,45.0,0.0));
#110=CARTESIAN_POINT('Trim point 1',(55.0,23.0,0.0));
#111=AXIS2_PLACEMENT_3D('Circle Placement',#112,#105,#106);
#112=CARTESIAN_POINT('Circle centre',(32.556,23.0,0.0));
#113=POLYLINE('Polyline for contour ',(#114,#115));
#114=CARTESIAN_POINT('Poly point 1',(37.0,45.0,0.0));
#115=CARTESIAN_POINT('Poly point 2',(11.0,45.0,0.0));
#116=TRIMMED_CURVE('Trimmed Curve',#117,(#118),(#119),.T.,.CARTESIAN.);
#117=CIRCLE('Circle1',#120,6.0);
#118=CARTESIAN_POINT('Trim point 1',(11.0,45.0,0.0));
#119=CARTESIAN_POINT('Trim point 2',(5.0,51.0,0.0));
#120=AXIS2_PLACEMENT_3D('Circle Placement',#121,#105,#106);
#121=CARTESIAN_POINT('Circle centre',(11.0,51.0,0.0));
#122=POLYLINE('Polyline for contour ',(#123,#124));
#123=CARTESIAN_POINT('Poly point 1',(5.0,51.0,0.0));
#124=CARTESIAN_POINT('Poly point 2',(5.0,95.0,0.0));
#125=AXIS2_PLACEMENT_3D('Axis for profile placement',#127,#128,#129);
#126=COMPOSITE_CURVE('E_LAB_A_OUT1 Curve',(#130,#131,#132,#133,#134,#135,#136,#137),.F.);
```

```
#127=CARTESIAN_POINT('point for profile placement',(0.0,0.0,0.0));
#128=DIRECTION('axis direction for profile placement',(0.0,0.0,1.0));
#129=DIRECTION('reference direction for profile placement',(1.0,0.0,0.0));
#130=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#138);
#131=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#146);
#132=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#149);
#133=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#155);
#134=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#158);
#135=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#164);
#136=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#167);
#137=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#173);
#138=TRIMMED_CURVE('Trimmed Curve',#139,(#140),(#141),.T.,.CARTESIAN.);
#139=CIRCLE('Circle1',#142,10.0);
#140=CARTESIAN_POINT('Trim point 1',(-13.0,103.0,0.0));
#141=CARTESIAN_POINT('Trim point 2',(-3.0,113.0,0.0));
#142=AXIS2_PLACEMENT_3D('Circle Placement',#143,#144,#145);
#143=CARTESIAN_POINT('Circle centre',(-3.0,103.0,0.0));
#144=DIRECTION('Z Direction',(0.0,0.0,1.0));
#145=DIRECTION('X Direction',(1.0,0.0,0.0));
#146=POLYLINE('Polyline for contour ',(#147,#148));
#147=CARTESIAN_POINT('Poly point 1',(-3.0,113.0,0.0));
#148=CARTESIAN_POINT('Poly point 2',(153.0,113.0,0.0));
#149=TRIMMED_CURVE('Trimmed Curve',#150,(#151),(#152),.T.,.CARTESIAN.);
#150=CIRCLE('Circle1',#153,10.0);
#151=CARTESIAN_POINT('Trim point 1',(153.0,113.0,0.0));
#152=CARTESIAN_POINT('Trim point 2',(163.0,103.0,0.0));
#153=AXIS2_PLACEMENT_3D('Circle Placement',#154,#144,#145);
#154=CARTESIAN_POINT('Circle centre',(153.0,103.0,0.0));
#155=POLYLINE('Polyline for contour ',(#156,#157));
#156=CARTESIAN_POINT('Poly point 1',(163.0,103.0,0.0));
#157=CARTESIAN_POINT('Poly point 2',(163.0,-3.0,0.0));
#158=TRIMMED_CURVE('Trimmed Curve',#159,(#160),(#161),.T.,.CARTESIAN.);
#159=CIRCLE('Circle1',#162,10.0);
#160=CARTESIAN_POINT('Trim point 1',(163.0,-3.0,0.0));
#161=CARTESIAN_POINT('Trim point 2',(153.0,-13.0,0.0));
#162=AXIS2_PLACEMENT_3D('Circle Placement',#163,#144,#145);
#163=CARTESIAN_POINT('Circle centre',(153.0,-3.0,0.0));
#164=POLYLINE('Polyline for contour ',(#165,#166));
#165=CARTESIAN_POINT('Poly point 1',(153.0,-13.0,0.0));
#166=CARTESIAN_POINT('Poly point 2',(-3.0,-13.0,0.0));
#167=TRIMMED_CURVE('Trimmed Curve',#168,(#169),(#170),.T.,.CARTESIAN.);
#168=CIRCLE('Circle1',#171,10.0);
#169=CARTESIAN_POINT('Trim point 1',(-3.0,-13.0,0.0));
#170=CARTESIAN_POINT('Trim point 2',(-13.0,-3.0,0.0));
#171=AXIS2_PLACEMENT_3D('Circle Placement',#172,#144,#145);
#172=CARTESIAN_POINT('Circle centre',(-3.0,-3.0,0.0));
#173=POLYLINE('Polyline for contour ',(#174,#175));
#174=CARTESIAN_POINT('Poly point 1',(-13.0,-3.0,0.0));
#175=CARTESIAN_POINT('Poly point 2',(-13.0,103.0,0.0));
#176=PLANE('Security plane for pocket',#179);
#177=CLOSED_POCKET('pocket',#3,(#178,#183),#184,#185,(#186),$,#187,#188,#189,#190);
#178=BOTTOM_AND_SIDE_ROUGH_MILLING($,$,'Rough milling for pocket',$,$,$,$,$,$,$,$,$,$,$,$,$);
#179=AXIS2_PLACEMENT_3D('Axis for Security axis placement',#180,#181,#182);
#180=CARTESIAN_POINT('point for Security axis placement',(0.0,0.0,10.0));
#181=DIRECTION('axis direction for Security axis placement',(0.0,0.0,1.0));
#182=DIRECTION('reference direction for Security axis placement',(1.0,0.0,0.0));
#183=BOTTOM_AND_SIDE_FINISH_MILLING($,$,'Rough milling for pocket',$,$,$,$,$,$,$,$,$,$,$,$,$);
#184=AXIS2_PLACEMENT_3D('Axis for pocket placement',#191,#192,#193);
#185=PLANE('Pocket Depth plane',#194);
#186=BOSS('pocket',#3,(#178,#183),#198,#199,#200,0.0);
#187=PLANAR_POCKET_BOTTOM_CONDITION();
#188=TOLERANCED_LENGTH_MEASURE(2.0,$);
#189=TOLERANCED_LENGTH_MEASURE(0.05,$);
#190=GENERAL_CLOSED_PROFILE(#276,#277);
#191=CARTESIAN_POINT('point for pocket placement',(0.0,0.0,0.0));
#192=DIRECTION('axis direction for pocket placement',(0.0,0.0,1.0));
#193=DIRECTION('reference direction for pocket placement',(1.0,0.0,0.0));
#194=AXIS2_PLACEMENT_3D('Axis for Depth placement',#195,#196,#197);
#195=CARTESIAN_POINT('point for Depth placement',(0.0,0.0,-15.0));
#196=DIRECTION('axis direction for Depth placement',(0.0,0.0,1.0));
```

```
#197=DIRECTION('reference direction for Depth placement',(1.0,0.0,0.0));
#198=AXIS2_PLACEMENT_3D('Axis for pocket placement',#201,#202,#203);
#199=PLANE('Pocket Depth plane',#204);
#200=GENERAL_CLOSED_PROFILE(#208,#209);
#201=CARTESIAN_POINT('point for pocket placement',(0.0,0.0,0.0));
#202=DIRECTION('axis direction for pocket placement',(0.0,0.0,1.0));
#203=DIRECTION('reference direction for pocket placement',(1.0,0.0,0.0));
#204=AXIS2_PLACEMENT_3D('Axis for Depth placement',#205,#206,#207);
#205=CARTESIAN_POINT('point for Depth placement',(0.0,0.0,-15.0));
#206=DIRECTION('axis direction for Depth placement',(0.0,0.0,1.0));
#207=DIRECTION('reference direction for Depth placement',(1.0,0.0,0.0));
#208=AXIS2_PLACEMENT_3D('Axis for profile placement',#210,#211,#212);
#209=COMPOSITE_CURVE('E_LAB_A_FIST_OUT
Curve',(#213,#214,#215,#216,#217,#218,#219,#220,#221,#222),.F.);
#210=CARTESIAN_POINT('point for profile placement',(0.0,0.0,0.0));
#211=DIRECTION('axis direction for profile placement',(0.0,0.0,1.0));
#212=DIRECTION('reference direction for profile placement',(1.0,0.0,0.0));
#213=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#223);
#214=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#231);
#215=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#237);
#216=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#243);
#217=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#249);
#218=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#255);
#219=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#258);
#220=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#264);
#221=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#267);
#222=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#273);
#223=TRIMMED_CURVE('Trimmed Curve',#224,(#225),(#226),.T.,.CARTESIAN.);
#224=CIRCLE('Circle1',#227,15.0);
#225=CARTESIAN_POINT('Trim point 1',(60.0,55.0,0.0));
#226=CARTESIAN_POINT('Trim point 2',(81.708,68.416,0.0));
#227=AXIS2_PLACEMENT_3D('Circle Placement',#228,#229,#230);
#228=CARTESIAN_POINT('Circle centre',(75.0,55.0,0.0));
#229=DIRECTION('Z Direction',(0.0,0.0,1.0));
#230=DIRECTION('X Direction',(1.0,0.0,0.0));
#231=TRIMMED_CURVE('Trimmed Curve',#232,(#233),(#234),.T.,.CARTESIAN.);
#232=CIRCLE('Circle1',#235,12.951305725678786);
#233=CARTESIAN_POINT('Trim point 2',(93.292,68.416,0.0));
#234=CARTESIAN_POINT('Trim point 1',(81.708,68.416,0.0));
#235=AXIS2_PLACEMENT_3D('Circle Placement',#236,#229,#230);
#236=CARTESIAN_POINT('Circle centre',(87.5,80.0,0.0));
#237=TRIMMED_CURVE('Trimmed Curve',#238,(#239),(#240),.T.,.CARTESIAN.);
#238=CIRCLE('Circle1',#241,14.999543993068587);
#239=CARTESIAN_POINT('Trim point 1',(93.292,68.416,0.0));
#240=CARTESIAN_POINT('Trim point 2',(106.708,68.416,0.0));
#241=AXIS2_PLACEMENT_3D('Circle Placement',#242,#229,#230);
#242=CARTESIAN_POINT('Circle centre',(100.0,55.0,0.0));
#243=TRIMMED_CURVE('Trimmed Curve',#244,(#245),(#246),.T.,.CARTESIAN.);
#244=CIRCLE('Circle1',#247,12.951305725678786);
#245=CARTESIAN_POINT('Trim point 2',(118.292,68.416,0.0));
#246=CARTESIAN_POINT('Trim point 1',(106.708,68.416,0.0));
#247=AXIS2_PLACEMENT_3D('Circle Placement',#248,#229,#230);
#248=CARTESIAN_POINT('Circle centre',(112.5,80.0,0.0));
#249=TRIMMED_CURVE('Trimmed Curve',#250,(#251),(#252),.T.,.CARTESIAN.);
#250=CIRCLE('Circle1',#253,14.999543993068587);
#251=CARTESIAN_POINT('Trim point 1',(118.292,68.416,0.0));
#252=CARTESIAN_POINT('Trim point 2',(140.0,55.0,0.0));
#253=AXIS2_PLACEMENT_3D('Circle Placement',#254,#229,#230);
#254=CARTESIAN_POINT('Circle centre',(125.0,55.0,0.0));
#255=POLYLINE('Polyline for contour ',(#256,#257));
#256=CARTESIAN_POINT('Poly point 1',(140.0,55.0,0.0));
#257=CARTESIAN_POINT('Poly point 2',(140.0,23.0,0.0));
#258=TRIMMED_CURVE('Trimmed Curve',#259,(#260),(#261),.T.,.CARTESIAN.);
#259=CIRCLE('Circle1',#262,13.0);
#260=CARTESIAN_POINT('Trim point 1',(140.0,23.0,0.0));
#261=CARTESIAN_POINT('Trim point 2',(127.0,10.0,0.0));
#262=AXIS2_PLACEMENT_3D('Circle Placement',#263,#229,#230);
#263=CARTESIAN_POINT('Circle centre',(127.0,23.0,0.0));
#264=POLYLINE('Polyline for contour ',(#265,#266));
#265=CARTESIAN_POINT('Poly point 1',(127.0,10.0,0.0));
```

```
#266=CARTESIAN_POINT('Poly point 2',(73.0,10.0,0.0));
#267=TRIMMED_CURVE('Trimmed Curve',#268,(#269),(#270),.T.,.CARTESIAN.);
#268=CIRCLE('Circle1',#271,13.0);
#269=CARTESIAN_POINT('Trim point 1',(73.0,10.0,0.0));
#270=CARTESIAN_POINT('Trim point 2',(60.0,23.0,0.0));
#271=AXIS2_PLACEMENT_3D('Circle Placement',#272,#229,#230);
#272=CARTESIAN_POINT('Circle centre',(73.0,23.0,0.0));
#273=POLYLINE('Polyline for contour ',(#274,#275));
#274=CARTESIAN_POINT('Poly point 1',(60.0,23.0,0.0));
#275=CARTESIAN_POINT('Poly point 2',(60.0,55.0,0.0));
#276=AXIS2_PLACEMENT_3D('Axis for profile placement',#278,#279,#280);
#277=COMPOSITE_CURVE('E_LAB_A_CLEAR_OUT
Curve',(#281,#282,#283,#284,#285,#286,#287,#288,#289,#290,#291),.F.);
#278=CARTESIAN_POINT('point for profile placement',(0.0,0.0,0.0));
#279=DIRECTION('axis direction for profile placement',(0.0,0.0,1.0));
#280=DIRECTION('reference direction for profile placement',(1.0,0.0,0.0));
#281=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#292);
#282=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#295);
#283=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#303);
#284=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#306);
#285=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#312);
#286=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#315);
#287=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#321);
#288=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#327);
#289=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#330);
#290=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#336);
#291=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#339);
#292=POLYLINE('Polyline for contour ',(#293,#294));
#293=CARTESIAN_POINT('Poly point 1',(0.0,108.0,0.0));
#294=CARTESIAN_POINT('Poly point 2',(150.0,108.0,0.0));
#295=TRIMMED_CURVE('Trimmed Curve',#296,(#297),(#298),.T.,.CARTESIAN.);
#296=CIRCLE('Circle1',#299,8.0);
#297=CARTESIAN_POINT('Trim point 1',(150.0,108.0,0.0));
#298=CARTESIAN_POINT('Trim point 2',(158.0,100.0,0.0));
#299=AXIS2_PLACEMENT_3D('Circle Placement',#300,#301,#302);
#300=CARTESIAN_POINT('Circle centre',(150.0,100.0,0.0));
#301=DIRECTION('Z Direction',(0.0,0.0,1.0));
#302=DIRECTION('X Direction',(1.0,0.0,0.0));
#303=POLYLINE('Polyline for contour ',(#304,#305));
#304=CARTESIAN_POINT('Poly point 1',(158.0,100.0,0.0));
#305=CARTESIAN_POINT('Poly point 2',(158.0,0.0,0.0));
#306=TRIMMED_CURVE('Trimmed Curve',#307,(#308),(#309),.T.,.CARTESIAN.);
#307=CIRCLE('Circle1',#310,8.0);
#308=CARTESIAN_POINT('Trim point 1',(158.0,0.0,0.0));
#309=CARTESIAN_POINT('Trim point 2',(150.0,-8.0,0.0));
#310=AXIS2_PLACEMENT_3D('Circle Placement',#311,#301,#302);
#311=CARTESIAN_POINT('Circle centre',(150.0,0.0,0.0));
#312=POLYLINE('Polyline for contour ',(#313,#314));
#313=CARTESIAN_POINT('Poly point 1',(150.0,-8.0,0.0));
#314=CARTESIAN_POINT('Poly point 2',(73.0,-8.0,0.0));
#315=TRIMMED_CURVE('Trimmed Curve',#316,(#317),(#318),.T.,.CARTESIAN.);
#316=CIRCLE('Circle1',#319,31.0);
#317=CARTESIAN_POINT('Trim point 1',(73.0,-8.0,0.0));
#318=CARTESIAN_POINT('Trim point 2',(42.0,23.0,0.0));
#319=AXIS2_PLACEMENT_3D('Circle Placement',#320,#301,#302);
#320=CARTESIAN_POINT('Circle centre',(73.0,23.0,0.0));
#321=TRIMMED_CURVE('Trimmed Curve',#322,(#323),(#324),.T.,.CARTESIAN.);
#322=CIRCLE('Circle1',#325,9.444000000000003);
#323=CARTESIAN_POINT('Trim point 2',(35.419,32.0,0.0));
#324=CARTESIAN_POINT('Trim point 1',(42.0,23.0,0.0));
#325=AXIS2_PLACEMENT_3D('Circle Placement',#326,#301,#302);
#326=CARTESIAN_POINT('Circle centre',(32.556,23.0,0.0));
#327=POLYLINE('Polyline for contour ',(#328,#329));
#328=CARTESIAN_POINT('Poly point 1',(35.419,32.0,0.0));
#329=CARTESIAN_POINT('Poly point 2',(0.0,32.0,0.0));
#330=TRIMMED_CURVE('Trimmed Curve',#331,(#332),(#333),.T.,.CARTESIAN.);
#331=CIRCLE('Circle1',#334,8.0);
#332=CARTESIAN_POINT('Trim point 1',(0.0,32.0,0.0));
#333=CARTESIAN_POINT('Trim point 2',(-8.0,40.0,0.0));
#334=AXIS2_PLACEMENT_3D('Circle Placement',#335,#301,#302);
```

```
#335=CARTESIAN_POINT('Circle centre',(0.0,40.0,0.0));
#336=POLYLINE('Polyline for contour ',(#337,#338));
#337=CARTESIAN_POINT('Poly point 1',(-8.0,40.0,0.0));
#338=CARTESIAN_POINT('Poly point 2',(-8.0,100.0,0.0));
#339=TRIMMED_CURVE('Trimmed Curve',#340,(#341),(#342),.T.,.CARTESIAN.);
#340=CIRCLE('Circle1',#343,8.0);
#341=CARTESIAN_POINT('Trim point 1',(-8.0,100.0,0.0));
#342=CARTESIAN_POINT('Trim point 2',(0.0,108.0,0.0));
#343=AXIS2_PLACEMENT_3D('Circle Placement',#344,#301,#302);
#344=CARTESIAN_POINT('Circle centre',(0.0,100.0,0.0));
#345=PLANE('Security plane for pocket',#348);
#346=CLOSED_POCKET('pocket',#3,(#347,#352),#353,#354,(),$,#355,#356,#357,#358);
#347=BOTTOM_AND_SIDE_ROUGH_MILLING($,$,'Rough milling for pocket',$,$,$,$,$,$,$,$,$,$,$,$,$);
#348=AXIS2_PLACEMENT_3D('Axis for Security axis placement',#349,#350,#351);
#349=CARTESIAN_POINT('point for Security axis placement',(0.0,0.0,10.0));
#350=DIRECTION('axis direction for Security axis placement',(0.0,0.0,1.0));
#351=DIRECTION('reference direction for Security axis placement',(1.0,0.0,0.0));
#352=BOTTOM_AND_SIDE_FINISH_MILLING($,$,'Rough milling for pocket',$,$,$,$,$,$,$,$,$,$,$,$,$);
#353=AXIS2_PLACEMENT_3D('Axis for pocket placement',#359,#360,#361);
#354=PLANE('Pocket Depth plane',#362);
#355=PLANAR_POCKET_BOTTOM_CONDITION();
#356=TOLERANCED_LENGTH_MEASURE(2.0,$);
#357=TOLERANCED_LENGTH_MEASURE(0.05,$);
#358=GENERAL_CLOSED_PROFILE(#366,#367);
#359=CARTESIAN_POINT('point for pocket placement',(0.0,0.0,0.0));
#360=DIRECTION('axis direction for pocket placement',(0.0,0.0,1.0));
#361=DIRECTION('reference direction for pocket placement',(1.0,0.0,0.0));
#362=AXIS2_PLACEMENT_3D('Axis for Depth placement',#363,#364,#365);
#363=CARTESIAN_POINT('point for Depth placement',(0.0,0.0,-10.0));
#364=DIRECTION('axis direction for Depth placement',(0.0,0.0,1.0));
#365=DIRECTION('reference direction for Depth placement',(1.0,0.0,0.0));
#366=AXIS2_PLACEMENT_3D('Axis for profile placement',#368,#369,#370);
#367=COMPOSITE_CURVE('E_LAB_A_FIST_IN
Curve',(#371,#372,#373,#374,#375,#376,#377,#378,#379,#380,#381,#382,#383,#384),.F.);
#368=CARTESIAN_POINT('point for profile placement',(0.0,0.0,0.0));
#369=DIRECTION('axis direction for profile placement',(0.0,0.0,1.0));
#370=DIRECTION('reference direction for profile placement',(1.0,0.0,0.0));
#371=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#385);
#372=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#393);
#373=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#399);
#374=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#405);
#375=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#411);
#376=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#417);
#377=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#420);
#378=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#426);
#379=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#429);
#380=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#432);
#381=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#435);
#382=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#441);
#383=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#444);
#384=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#447);
#385=TRIMMED_CURVE('Trimmed Curve',#386,(#387),(#388),.T.,.CARTESIAN.);
#386=CIRCLE('Circle1',#389,10.002000449910003);
#387=CARTESIAN_POINT('Trim point 1',(65.0,55.0,0.0));
#388=CARTESIAN_POINT('Trim point 2',(79.474,63.943,0.0));
#389=AXIS2_PLACEMENT_3D('Circle Placement',#390,#391,#392);
#390=CARTESIAN_POINT('Circle centre',(75.002,54.997,0.0));
#391=DIRECTION('Z Direction',(0.0,0.0,1.0));
#392=DIRECTION('X Direction',(1.0,0.0,0.0));
#393=TRIMMED_CURVE('Trimmed Curve',#394,(#395),(#396),.T.,.CARTESIAN.);
#394=CIRCLE('Circle1',#397,17.951153862635124);
#395=CARTESIAN_POINT('Trim point 2',(95.526,63.943,0.0));
#396=CARTESIAN_POINT('Trim point 1',(79.474,63.943,0.0));
#397=AXIS2_PLACEMENT_3D('Circle Placement',#398,#391,#392);
#398=CARTESIAN_POINT('Circle centre',(87.5,80.0,0.0));
#399=TRIMMED_CURVE('Trimmed Curve',#400,(#401),(#402),.T.,.CARTESIAN.);
#400=CIRCLE('Circle1',#403,10.005957025692243);
#401=CARTESIAN_POINT('Trim point 1',(95.526,63.943,0.0));
#402=CARTESIAN_POINT('Trim point 2',(104.474,63.943,0.0));
#403=AXIS2_PLACEMENT_3D('Circle Placement',#404,#391,#392);
```

```
#404=CARTESIAN_POINT('Circle centre',(100.0,54.993,0.0));
#405=TRIMMED_CURVE('Trimmed Curve',#406,(#407),(#408),.T.,.CARTESIAN.);
#406=CIRCLE('Circle1',#409,17.940420870202573);
#407=CARTESIAN_POINT('Trim point 2',(120.526,63.943,0.0));
#408=CARTESIAN_POINT('Trim point 1',(104.474,63.943,0.0));
#409=AXIS2_PLACEMENT_3D('Circle Placement',#410,#391,#392);
#410=CARTESIAN_POINT('Circle centre',(112.5,79.988,0.0));
#411=TRIMMED_CURVE('Trimmed Curve',#412,(#413),(#414),.T.,.CARTESIAN.);
#412=CIRCLE('Circle1',#415,9.999696245386657);
#413=CARTESIAN_POINT('Trim point 1',(120.526,63.943,0.0));
#414=CARTESIAN_POINT('Trim point 2',(135.0,55.0,0.0));
#415=AXIS2_PLACEMENT_3D('Circle Placement',#416,#391,#392);
#416=CARTESIAN_POINT('Circle centre',(125.0,55.0,0.0));
#417=POLYLINE('Polyline for contour ',(#418,#419));
#418=CARTESIAN_POINT('Poly point 1',(135.0,55.0,0.0));
#419=CARTESIAN_POINT('Poly point 2',(135.0,23.0,0.0));
#420=TRIMMED_CURVE('Trimmed Curve',#421,(#422),(#423),.T.,.CARTESIAN.);
#421=CIRCLE('Circle1',#424,8.0);
#422=CARTESIAN_POINT('Trim point 1',(135.0,23.0,0.0));
#423=CARTESIAN_POINT('Trim point 2',(127.0,15.0,0.0));
#424=AXIS2_PLACEMENT_3D('Circle Placement',#425,#391,#392);
#425=CARTESIAN_POINT('Circle centre',(127.0,23.0,0.0));
#426=POLYLINE('Polyline for contour ',(#427,#428));
#427=CARTESIAN_POINT('Poly point 1',(127.0,15.0,0.0));
#428=CARTESIAN_POINT('Poly point 2',(118.0,15.0,0.0));
#429=POLYLINE('Polyline for contour ',(#430,#431));
#430=CARTESIAN_POINT('Poly point 1',(118.0,15.0,0.0));
#431=CARTESIAN_POINT('Poly point 2',(94.0,15.0,0.0));
#432=POLYLINE('Polyline for contour ',(#433,#434));
#433=CARTESIAN_POINT('Poly point 1',(94.0,15.0,0.0));
#434=CARTESIAN_POINT('Poly point 2',(73.0,15.0,0.0));
#435=TRIMMED_CURVE('Trimmed Curve',#436,(#437),(#438),.T.,.CARTESIAN.);
#436=CIRCLE('Circle1',#439,8.0);
#437=CARTESIAN_POINT('Trim point 1',(73.0,15.0,0.0));
#438=CARTESIAN_POINT('Trim point 2',(65.0,23.0,0.0));
#439=AXIS2_PLACEMENT_3D('Circle Placement',#440,#391,#392);
#440=CARTESIAN_POINT('Circle centre',(73.0,23.0,0.0));
#441=POLYLINE('Polyline for contour ',(#442,#443));
#442=CARTESIAN_POINT('Poly point 1',(65.0,23.0,0.0));
#443=CARTESIAN_POINT('Poly point 2',(65.0,35.0,0.0));
#444=POLYLINE('Polyline for contour ',(#445,#446));
#445=CARTESIAN_POINT('Poly point 1',(65.0,35.0,0.0));
#446=CARTESIAN_POINT('Poly point 2',(65.0,47.0,0.0));
#447=POLYLINE('Polyline for contour ',(#448,#449));
#448=CARTESIAN_POINT('Poly point 1',(65.0,47.0,0.0));
#449=CARTESIAN_POINT('Poly point 2',(65.0,55.0,0.0));
#450=PLANE('Security plane for pocket',#453);
#451=CLOSED_POCKET('pocket',#3,(#452,#457),#458,#459,(),$,#460,#461,#462,#463);
#452=BOTTOM_AND_SIDE_ROUGH_MILLING($,$,'Rough milling for pocket',$,$,$,$,$,$,$,$,$,$,$,$,$);
#453=AXIS2_PLACEMENT_3D('Axis for Security axis placement',#454,#455,#456);
#454=CARTESIAN_POINT('point for Security axis placement',(0.0,0.0,10.0));
#455=DIRECTION('axis direction for Security axis placement',(0.0,0.0,1.0));
#456=DIRECTION('reference direction for Security axis placement',(1.0,0.0,0.0));
#457=BOTTOM_AND_SIDE_FINISH_MILLING($,$,'Rough milling for pocket',$,$,$,$,$,$,$,$,$,$,$,$,$);
#458=AXIS2_PLACEMENT_3D('Axis for pocket placement',#464,#465,#466);
#459=PLANE('Pocket Depth plane',#467);
#460=PLANAR_POCKET_BOTTOM_CONDITION();
#461=TOLERANCED_LENGTH_MEASURE(2.0,$);
#462=TOLERANCED_LENGTH_MEASURE(0.05,$);
#463=GENERAL_CLOSED_PROFILE(#471,#472);
#464=CARTESIAN_POINT('point for pocket placement',(0.0,0.0,0.0));
#465=DIRECTION('axis direction for pocket placement',(0.0,0.0,1.0));
#466=DIRECTION('reference direction for pocket placement',(1.0,0.0,0.0));
#467=AXIS2_PLACEMENT_3D('Axis for Depth placement',#468,#469,#470);
#468=CARTESIAN_POINT('point for Depth placement',(0.0,0.0,-15.0));
#469=DIRECTION('axis direction for Depth placement',(0.0,0.0,1.0));
#470=DIRECTION('reference direction for Depth placement',(1.0,0.0,0.0));
#471=AXIS2_PLACEMENT_3D('Axis for profile placement',#473,#474,#475);
#472=COMPOSITE_CURVE('E_LAB_A_PRIME
Curve',(#476,#477,#478,#479,#480,#481,#482,#483,#484,#485,#486,#487,#488,#489,#490,#491),.F.);
```

```
#473=CARTESIAN_POINT('point for profile placement',(0.0,0.0,0.0));
#474=DIRECTION('axis direction for profile placement',(0.0,0.0,1.0));
#475=DIRECTION('reference direction for profile placement',(1.0,0.0,0.0));
#476=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#492);
#477=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#495);
#478=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#503);
#479=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#506);
#480=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#512);
#481=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#518);
#482=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#524);
#483=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#530);
#484=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#536);
#485=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#539);
#486=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#545);
#487=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#548);
#488=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#554);
#489=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#560);
#490=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#566);
#491=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#569);
#492=POLYLINE('Polyline for contour ',(#493,#494));
#493=CARTESIAN_POINT('Poly point 1',(100.0,42.0,0.0));
#494=CARTESIAN_POINT('Poly point 2',(70.0,42.0,0.0));
#495=TRIMMED_CURVE('Trimmed Curve',#496,(#497),(#498),.T.,.CARTESIAN.);
#496=CIRCLE('Circle1',#499,5.0);
#497=CARTESIAN_POINT('Trim point 1',(70.0,42.0,0.0));
#498=CARTESIAN_POINT('Trim point 2',(65.0,47.0,0.0));
#499=AXIS2_PLACEMENT_3D('Circle Placement',#500,#501,#502);
#500=CARTESIAN_POINT('Circle centre',(70.0,47.0,0.0));
#501=DIRECTION('Z Direction',(0.0,0.0,1.0));
#502=DIRECTION('X Direction',(1.0,0.0,0.0));
#503=POLYLINE('Polyline for contour ',(#504,#505));
#504=CARTESIAN_POINT('Poly point 1',(65.0,47.0,0.0));
#505=CARTESIAN_POINT('Poly point 2',(65.0,55.0,0.0));
#506=TRIMMED_CURVE('Trimmed Curve',#507,(#508),(#509),.T.,.CARTESIAN.);
#507=CIRCLE('Circle1',#510,10.0020004499100003);
#508=CARTESIAN_POINT('Trim point 1',(65.0,55.0,0.0));
#509=CARTESIAN_POINT('Trim point 2',(79.474,63.943,0.0));
#510=AXIS2_PLACEMENT_3D('Circle Placement',#511,#501,#502);
#511=CARTESIAN_POINT('Circle centre',(75.002,54.997,0.0));
#512=TRIMMED_CURVE('Trimmed Curve',#513,(#514),(#515),.T.,.CARTESIAN.);
#513=CIRCLE('Circle1',#516,17.951153862635124);
#514=CARTESIAN_POINT('Trim point 2',(95.526,63.943,0.0));
#515=CARTESIAN_POINT('Trim point 1',(79.474,63.943,0.0));
#516=AXIS2_PLACEMENT_3D('Circle Placement',#517,#501,#502);
#517=CARTESIAN_POINT('Circle centre',(87.5,80.0,0.0));
#518=TRIMMED_CURVE('Trimmed Curve',#519,(#520),(#521),.T.,.CARTESIAN.);
#519=CIRCLE('Circle1',#522,10.005957025692243);
#520=CARTESIAN_POINT('Trim point 1',(95.526,63.943,0.0));
#521=CARTESIAN_POINT('Trim point 2',(104.474,63.943,0.0));
#522=AXIS2_PLACEMENT_3D('Circle Placement',#523,#501,#502);
#523=CARTESIAN_POINT('Circle centre',(100.0,54.993,0.0));
#524=TRIMMED_CURVE('Trimmed Curve',#525,(#526),(#527),.T.,.CARTESIAN.);
#525=CIRCLE('Circle1',#528,17.940420870202573);
#526=CARTESIAN_POINT('Trim point 2',(120.526,63.943,0.0));
#527=CARTESIAN_POINT('Trim point 1',(104.474,63.943,0.0));
#528=AXIS2_PLACEMENT_3D('Circle Placement',#529,#501,#502);
#529=CARTESIAN_POINT('Circle centre',(112.5,79.988,0.0));
#530=TRIMMED_CURVE('Trimmed Curve',#531,(#532),(#533),.T.,.CARTESIAN.);
#531=CIRCLE('Circle1',#534,9.999696245386657);
#532=CARTESIAN_POINT('Trim point 1',(120.526,63.943,0.0));
#533=CARTESIAN_POINT('Trim point 2',(135.0,55.0,0.0));
#534=AXIS2_PLACEMENT_3D('Circle Placement',#535,#501,#502);
#535=CARTESIAN_POINT('Circle centre',(125.0,55.0,0.0));
#536=POLYLINE('Polyline for contour ',(#537,#538));
#537=CARTESIAN_POINT('Poly point 1',(135.0,55.0,0.0));
#538=CARTESIAN_POINT('Poly point 2',(135.0,23.0,0.0));
#539=TRIMMED_CURVE('Trimmed Curve',#540,(#541),(#542),.T.,.CARTESIAN.);
#540=CIRCLE('Circle1',#543,8.0);
#541=CARTESIAN_POINT('Trim point 1',(135.0,23.0,0.0));
#542=CARTESIAN_POINT('Trim point 2',(127.0,15.0,0.0));
```

```
#543=AXIS2_PLACEMENT_3D('Circle Placement',#544,#501,#502);
#544=CARTESIAN_POINT('Circle centre',(127.0,23.0,0.0));
#545=POLYLINE('Polyline for contour ',(#546,#547));
#546=CARTESIAN_POINT('Poly point 1',(127.0,15.0,0.0));
#547=CARTESIAN_POINT('Poly point 2',(118.0,15.0,0.0));
#548=TRIMMED_CURVE('Trimmed Curve',#549,(#550),(#551),.T.,.CARTESIAN.);
#549=CIRCLE('Circle1',#552,6.0);
#550=CARTESIAN_POINT('Trim point 1',(118.0,15.0,0.0));
#551=CARTESIAN_POINT('Trim point 2',(112.0,21.0,0.0));
#552=AXIS2_PLACEMENT_3D('Circle Placement',#553,#501,#502);
#553=CARTESIAN_POINT('Circle centre',(118.0,21.0,0.0));
#554=TRIMMED_CURVE('Trimmed Curve',#555,(#556),(#557),.T.,.CARTESIAN.);
#555=CIRCLE('Circle1',#558,6.0);
#556=CARTESIAN_POINT('Trim point 2',(109.5,25.873,0.0));
#557=CARTESIAN_POINT('Trim point 1',(112.0,21.0,0.0));
#558=AXIS2_PLACEMENT_3D('Circle Placement',#559,#501,#502);
#559=CARTESIAN_POINT('Circle centre',(106.0,21.0,0.0));
#560=TRIMMED_CURVE('Trimmed Curve',#561,(#562),(#563),.T.,.CARTESIAN.);
#561=CIRCLE('Circle1',#564,6.000489646687176);
#562=CARTESIAN_POINT('Trim point 1',(109.5,25.873,0.0));
#563=CARTESIAN_POINT('Trim point 2',(107.0,30.747,0.0));
#564=AXIS2_PLACEMENT_3D('Circle Placement',#565,#501,#502);
#565=CARTESIAN_POINT('Circle centre',(113.0,30.747,0.0));
#566=POLYLINE('Polyline for contour ',(#567,#568));
#567=CARTESIAN_POINT('Poly point 1',(107.0,30.747,0.0));
#568=CARTESIAN_POINT('Poly point 2',(107.0,35.0,0.0));
#569=TRIMMED_CURVE('Trimmed Curve',#570,(#571),(#572),.T.,.CARTESIAN.);
#570=CIRCLE('Circle1',#573,7.0);
#571=CARTESIAN_POINT('Trim point 2',(100.0,42.0,0.0));
#572=CARTESIAN_POINT('Trim point 1',(107.0,35.0,0.0));
#573=AXIS2_PLACEMENT_3D('Circle Placement',#574,#501,#502);
#574=CARTESIAN_POINT('Circle centre',(100.0,35.0,0.0));
#575=PLANE('Security plane for pocket',#578);
#576=CLOSED_POCKET('pocket',#3,(#577,#582),#583,#584,(),$,#585,#586,#587,#588);
#577=BOTTOM_AND_SIDE_ROUGH_MILLING($,$,'Rough milling for pocket',$,$,$,$,$,$,$,$,$,$,$,$,$,$);
#578=AXIS2_PLACEMENT_3D('Axis for Security axis placement',#579,#580,#581);
#579=CARTESIAN_POINT('point for Security axis placement',(0.0,0.0,10.0));
#580=DIRECTION('axis direction for Security axis placement',(0.0,0.0,1.0));
#581=DIRECTION('reference direction for Security axis placement',(1.0,0.0,0.0));
#582=BOTTOM_AND_SIDE_FINISH_MILLING($,$,'Rough milling for pocket',$,$,$,$,$,$,$,$,$,$,$,$,$,$);
#583=AXIS2_PLACEMENT_3D('Axis for pocket placement',#589,#590,#591);
#584=PLANE('Pocket Depth plane',#592);
#585=PLANAR_POCKET_BOTTOM_CONDITION();
#586=TOLERANCED_LENGTH_MEASURE(2.0,$);
#587=TOLERANCED_LENGTH_MEASURE(0.05,$);
#588=GENERAL_CLOSED_PROFILE(#596,#597);
#589=CARTESIAN_POINT('point for pocket placement',(0.0,0.0,0.0));
#590=DIRECTION('axis direction for pocket placement',(0.0,0.0,1.0));
#591=DIRECTION('reference direction for pocket placement',(1.0,0.0,0.0));
#592=AXIS2_PLACEMENT_3D('Axis for Depth placement',#593,#594,#595);
#593=CARTESIAN_POINT('point for Depth placement',(0.0,0.0,-15.0));
#594=DIRECTION('axis direction for Depth placement',(0.0,0.0,1.0));
#595=DIRECTION('reference direction for Depth placement',(1.0,0.0,0.0));
#596=AXIS2_PLACEMENT_3D('Axis for profile placement',#598,#599,#600);
#597=COMPOSITE_CURVE('E_LAB_A_SECOND
Curve',(#601,#602,#603,#604,#605,#606,#607,#608,#609,#610),.F.);
#598=CARTESIAN_POINT('point for profile placement',(0.0,0.0,0.0));
#599=DIRECTION('axis direction for profile placement',(0.0,0.0,1.0));
#600=DIRECTION('reference direction for profile placement',(1.0,0.0,0.0));
#601=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#611);
#602=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#614);
#603=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#622);
#604=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#625);
#605=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#631);
#606=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#637);
#607=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#643);
#608=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#646);
#609=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#652);
#610=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#655);
#611=POLYLINE('Polyline for contour ',(#612,#613));
```

```
#612=CARTESIAN_POINT('Poly point 1',(70.0,40.0,0.0));
#613=CARTESIAN_POINT('Poly point 2',(100.0,40.0,0.0));
#614=TRIMMED_CURVE('Trimmed Curve',#615,(#616),(#617),.T.,.CARTESIAN.);
#615=CIRCLE('Circle1',#618,5.0);
#616=CARTESIAN_POINT('Trim point 1',(100.0,40.0,0.0));
#617=CARTESIAN_POINT('Trim point 2',(105.0,35.0,0.0));
#618=AXIS2_PLACEMENT_3D('Circle Placement',#619,#620,#621);
#619=CARTESIAN_POINT('Circle centre',(100.0,35.0,0.0));
#620=DIRECTION('Z Direction',(0.0,0.0,1.0));
#621=DIRECTION('X Direction',(1.0,0.0,0.0));
#622=POLYLINE('Polyline for contour ',(#623,#624));
#623=CARTESIAN_POINT('Poly point 1',(105.0,35.0,0.0));
#624=CARTESIAN_POINT('Poly point 2',(105.0,28.746,0.0));
#625=TRIMMED_CURVE('Trimmed Curve',#626,(#627),(#628),.T.,.CARTESIAN.);
#626=CIRCLE('Circle1',#629,2.5);
#627=CARTESIAN_POINT('Trim point 1',(105.0,28.746,0.0));
#628=CARTESIAN_POINT('Trim point 2',(103.529,26.468,0.0));
#629=AXIS2_PLACEMENT_3D('Circle Placement',#630,#620,#621);
#630=CARTESIAN_POINT('Circle centre',(102.5,28.746,0.0));
#631=TRIMMED_CURVE('Trimmed Curve',#632,(#633),(#634),.T.,.CARTESIAN.);
#632=CIRCLE('Circle1',#635,6.000405402970704);
#633=CARTESIAN_POINT('Trim point 2',(100.0,21.0,0.0));
#634=CARTESIAN_POINT('Trim point 1',(103.529,26.468,0.0));
#635=AXIS2_PLACEMENT_3D('Circle Placement',#636,#620,#621);
#636=CARTESIAN_POINT('Circle centre',(106.0,21.0,0.0));
#637=TRIMMED_CURVE('Trimmed Curve',#638,(#639),(#640),.T.,.CARTESIAN.);
#638=CIRCLE('Circle1',#641,6.0);
#639=CARTESIAN_POINT('Trim point 1',(100.0,21.0,0.0));
#640=CARTESIAN_POINT('Trim point 2',(94.0,15.0,0.0));
#641=AXIS2_PLACEMENT_3D('Circle Placement',#642,#620,#621);
#642=CARTESIAN_POINT('Circle centre',(94.0,21.0,0.0));
#643=POLYLINE('Polyline for contour ',(#644,#645));
#644=CARTESIAN_POINT('Poly point 1',(94.0,15.0,0.0));
#645=CARTESIAN_POINT('Poly point 2',(73.0,15.0,0.0));
#646=TRIMMED_CURVE('Trimmed Curve',#647,(#648),(#649),.T.,.CARTESIAN.);
#647=CIRCLE('Circle1',#650,8.0);
#648=CARTESIAN_POINT('Trim point 1',(73.0,15.0,0.0));
#649=CARTESIAN_POINT('Trim point 2',(65.0,23.0,0.0));
#650=AXIS2_PLACEMENT_3D('Circle Placement',#651,#620,#621);
#651=CARTESIAN_POINT('Circle centre',(73.0,23.0,0.0));
#652=POLYLINE('Polyline for contour ',(#653,#654));
#653=CARTESIAN_POINT('Poly point 1',(65.0,23.0,0.0));
#654=CARTESIAN_POINT('Poly point 2',(65.0,35.0,0.0));
#655=TRIMMED_CURVE('Trimmed Curve',#656,(#657),(#658),.T.,.CARTESIAN.);
#656=CIRCLE('Circle1',#659,5.0);
#657=CARTESIAN_POINT('Trim point 1',(65.0,35.0,0.0));
#658=CARTESIAN_POINT('Trim point 2',(70.0,40.0,0.0));
#659=AXIS2_PLACEMENT_3D('Circle Placement',#660,#620,#621);
#660=CARTESIAN_POINT('Circle centre',(70.0,35.0,0.0));
#661=PLANE('Drilling plane',#664);
#662=ROUND_HOLE('smallhole1',#3,(#663),#668,#669,#670,$,#671);
#663=DRILLING($,$,'Drilling',$,$,$,$,$,$,$,$,$,$);
#664=AXIS2_PLACEMENT_3D('Axis for Drilling plane placement',#665,#666,#667);
#665=CARTESIAN_POINT('point for Drilling plane placement',(0.0,0.0,10.0));
#666=DIRECTION('axis direction for Drilling plane placement',(0.0,0.0,1.0));
#667=DIRECTION('reference direction for Drilling plane placement',(1.0,0.0,0.0));
#668=AXIS2_PLACEMENT_3D('Axis for Placement for Hole',#672,#673,#674);
#669=PLANE('Hole Depth Surface',#675);
#670=TOLERANCED_LENGTH_MEASURE(3.0,$);
#671=THROUGH_BOTTOM_CONDITION();
#672=CARTESIAN_POINT('point for Placement for Hole',(46.5,57.5,0.0));
#673=DIRECTION('axis direction for Placement for Hole',(0.0,0.0,1.0));
#674=DIRECTION('reference direction for Placement for Hole',(1.0,0.0,0.0));
#675=AXIS2_PLACEMENT_3D('Axis for Placement for Hole Depth',#676,#677,#678);
#676=CARTESIAN_POINT('point for Placement for Hole Depth',(0.0,0.0,-40.0));
#677=DIRECTION('axis direction for Placement for Hole Depth',(0.0,0.0,1.0));
#678=DIRECTION('reference direction for Placement for Hole Depth',(1.0,0.0,0.0));
#679=PLANE('Drilling plane',#682);
#680=ROUND_HOLE('smallhole2',#3,(#681),#686,#687,#688,$,#689);
#681=DRILLING($,$,'Drilling',$,$,$,$,$,$,$,$,$,$);
```

```
#682=AXIS2_PLACEMENT_3D('Axis for Drilling plane placement',#683,#684,#685);
#683=CARTESIAN_POINT('point for Drilling plane placement',(0.0,0.0,10.0));
#684=DIRECTION('axis direction for Drilling plane placement',(0.0,0.0,1.0));
#685=DIRECTION('reference direction for Drilling plane placement',(1.0,0.0,0.0));
#686=AXIS2_PLACEMENT_3D('Axis for Placement for Hole',#690,#691,#692);
#687=PLANE('Hole Depth Surface',#693);
#688=TOLERANCED_LENGTH_MEASURE(3.0,$);
#689=THROUGH_BOTTOM_CONDITION();
#690=CARTESIAN_POINT('point for Placement for Hole',(46.5,62.5,0.0));
#691=DIRECTION('axis direction for Placement for Hole',(0.0,0.0,1.0));
#692=DIRECTION('reference direction for Placement for Hole',(1.0,0.0,0.0));
#693=AXIS2_PLACEMENT_3D('Axis for Placement for Hole Depth',#694,#695,#696);
#694=CARTESIAN_POINT('point for Placement for Hole Depth',(0.0,0.0,-40.0));
#695=DIRECTION('axis direction for Placement for Hole Depth',(0.0,0.0,1.0));
#696=DIRECTION('reference direction for Placement for Hole Depth',(1.0,0.0,0.0));
#697=PLANE('Drilling plane',#700);
#698=ROUND_HOLE('smallhole3',#3,(#699),#704,#705,#706,$,#707);
#699=DRILLING($,$,'Drilling',$,$,$,$,$,$,$,$,$,$,$);
#700=AXIS2_PLACEMENT_3D('Axis for Drilling plane placement',#701,#702,#703);
#701=CARTESIAN_POINT('point for Drilling plane placement',(0.0,0.0,10.0));
#702=DIRECTION('axis direction for Drilling plane placement',(0.0,0.0,1.0));
#703=DIRECTION('reference direction for Drilling plane placement',(1.0,0.0,0.0));
#704=AXIS2_PLACEMENT_3D('Axis for Placement for Hole',#708,#709,#710);
#705=PLANE('Hole Depth Surface',#711);
#706=TOLERANCED_LENGTH_MEASURE(3.0,$);
#707=THROUGH_BOTTOM_CONDITION();
#708=CARTESIAN_POINT('point for Placement for Hole',(46.5,67.5,0.0));
#709=DIRECTION('axis direction for Placement for Hole',(0.0,0.0,1.0));
#710=DIRECTION('reference direction for Placement for Hole',(1.0,0.0,0.0));
#711=AXIS2_PLACEMENT_3D('Axis for Placement for Hole Depth',#712,#713,#714);
#712=CARTESIAN_POINT('point for Placement for Hole Depth',(0.0,0.0,-40.0));
#713=DIRECTION('axis direction for Placement for Hole Depth',(0.0,0.0,1.0));
#714=DIRECTION('reference direction for Placement for Hole Depth',(1.0,0.0,0.0));
#715=PLANE('Drilling plane',#718);
#716=ROUND_HOLE('smallhole4',#3,(#717),#722,#723,#724,$,#725);
#717=DRILLING($,$,'Drilling',$,$,$,$,$,$,$,$,$,$,$);
#718=AXIS2_PLACEMENT_3D('Axis for Drilling plane placement',#719,#720,#721);
#719=CARTESIAN_POINT('point for Drilling plane placement',(0.0,0.0,10.0));
#720=DIRECTION('axis direction for Drilling plane placement',(0.0,0.0,1.0));
#721=DIRECTION('reference direction for Drilling plane placement',(1.0,0.0,0.0));
#722=AXIS2_PLACEMENT_3D('Axis for Placement for Hole',#726,#727,#728);
#723=PLANE('Hole Depth Surface',#729);
#724=TOLERANCED_LENGTH_MEASURE(3.0,$);
#725=THROUGH_BOTTOM_CONDITION();
#726=CARTESIAN_POINT('point for Placement for Hole',(46.5,72.5,0.0));
#727=DIRECTION('axis direction for Placement for Hole',(0.0,0.0,1.0));
#728=DIRECTION('reference direction for Placement for Hole',(1.0,0.0,0.0));
#729=AXIS2_PLACEMENT_3D('Axis for Placement for Hole Depth',#730,#731,#732);
#730=CARTESIAN_POINT('point for Placement for Hole Depth',(0.0,0.0,-40.0));
#731=DIRECTION('axis direction for Placement for Hole Depth',(0.0,0.0,1.0));
#732=DIRECTION('reference direction for Placement for Hole Depth',(1.0,0.0,0.0));
#733=PLANE('Drilling plane',#736);
#734=ROUND_HOLE('smallhole5',#3,(#735),#740,#741,#742,$,#743);
#735=DRILLING($,$,'Drilling',$,$,$,$,$,$,$,$,$,$,$);
#736=AXIS2_PLACEMENT_3D('Axis for Drilling plane placement',#737,#738,#739);
#737=CARTESIAN_POINT('point for Drilling plane placement',(0.0,0.0,10.0));
#738=DIRECTION('axis direction for Drilling plane placement',(0.0,0.0,1.0));
#739=DIRECTION('reference direction for Drilling plane placement',(1.0,0.0,0.0));
#740=AXIS2_PLACEMENT_3D('Axis for Placement for Hole',#744,#745,#746);
#741=PLANE('Hole Depth Surface',#747);
#742=TOLERANCED_LENGTH_MEASURE(3.0,$);
#743=THROUGH_BOTTOM_CONDITION();
#744=CARTESIAN_POINT('point for Placement for Hole',(46.5,77.5,0.0));
#745=DIRECTION('axis direction for Placement for Hole',(0.0,0.0,1.0));
#746=DIRECTION('reference direction for Placement for Hole',(1.0,0.0,0.0));
#747=AXIS2_PLACEMENT_3D('Axis for Placement for Hole Depth',#748,#749,#750);
#748=CARTESIAN_POINT('point for Placement for Hole Depth',(0.0,0.0,-40.0));
#749=DIRECTION('axis direction for Placement for Hole Depth',(0.0,0.0,1.0));
#750=DIRECTION('reference direction for Placement for Hole Depth',(1.0,0.0,0.0));
#751=PLANE('Drilling plane',#754);
```

```
#752=ROUND_HOLE('hole1',#3,(#753),#758,#759,#760,$,#761);
#753=DRILLING($,$,'Drilling',$,$,$,$,$,$,$,$,$,$,$);
#754=AXIS2_PLACEMENT_3D('Axis for Drilling plane placement',#755,#756,#757);
#755=CARTESIAN_POINT('point for Drilling plane placement',(0.0,0.0,10.0));
#756=DIRECTION('axis direction for Drilling plane placement',(0.0,0.0,1.0));
#757=DIRECTION('reference direction for Drilling plane placement',(1.0,0.0,0.0));
#758=AXIS2_PLACEMENT_3D('Axis for Placement for Hole',#762,#763,#764);
#759=PLANE('Hole Depth Surface',#765);
#760=TOLERANCED_LENGTH_MEASURE(8.0,$);
#761=THROUGH_BOTTOM_CONDITION();
#762=CARTESIAN_POINT('point for Placement for Hole',(136.0,86.0,0.0));
#763=DIRECTION('axis direction for Placement for Hole',(0.0,0.0,1.0));
#764=DIRECTION('reference direction for Placement for Hole',(1.0,0.0,0.0));
#765=AXIS2_PLACEMENT_3D('Axis for Placement for Hole Depth',#766,#767,#768);
#766=CARTESIAN_POINT('point for Placement for Hole Depth',(0.0,0.0,-40.0));
#767=DIRECTION('axis direction for Placement for Hole Depth',(0.0,0.0,1.0));
#768=DIRECTION('reference direction for Placement for Hole Depth',(1.0,0.0,0.0));
#769=PLANE('Drilling plane',#772);
#770=ROUND_HOLE('hole2',#3,(#771),#776,#777,#778,$,#779);
#771=DRILLING($,$,'Drilling',$,$,$,$,$,$,$,$,$,$,$);
#772=AXIS2_PLACEMENT_3D('Axis for Drilling plane placement',#773,#774,#775);
#773=CARTESIAN_POINT('point for Drilling plane placement',(0.0,0.0,10.0));
#774=DIRECTION('axis direction for Drilling plane placement',(0.0,0.0,1.0));
#775=DIRECTION('reference direction for Drilling plane placement',(1.0,0.0,0.0));
#776=AXIS2_PLACEMENT_3D('Axis for Placement for Hole',#780,#781,#782);
#777=PLANE('Hole Depth Surface',#783);
#778=TOLERANCED_LENGTH_MEASURE(8.0,$);
#779=THROUGH_BOTTOM_CONDITION();
#780=CARTESIAN_POINT('point for Placement for Hole',(112.5,80.0,0.0));
#781=DIRECTION('axis direction for Placement for Hole',(0.0,0.0,1.0));
#782=DIRECTION('reference direction for Placement for Hole',(1.0,0.0,0.0));
#783=AXIS2_PLACEMENT_3D('Axis for Placement for Hole Depth',#784,#785,#786);
#784=CARTESIAN_POINT('point for Placement for Hole Depth',(0.0,0.0,-40.0));
#785=DIRECTION('axis direction for Placement for Hole Depth',(0.0,0.0,1.0));
#786=DIRECTION('reference direction for Placement for Hole Depth',(1.0,0.0,0.0));
#787=PLANE('Drilling plane',#790);
#788=ROUND_HOLE('hole3',#3,(#789),#794,#795,#796,$,#797);
#789=DRILLING($,$,'Drilling',$,$,$,$,$,$,$,$,$,$,$);
#790=AXIS2_PLACEMENT_3D('Axis for Drilling plane placement',#791,#792,#793);
#791=CARTESIAN_POINT('point for Drilling plane placement',(0.0,0.0,10.0));
#792=DIRECTION('axis direction for Drilling plane placement',(0.0,0.0,1.0));
#793=DIRECTION('reference direction for Drilling plane placement',(1.0,0.0,0.0));
#794=AXIS2_PLACEMENT_3D('Axis for Placement for Hole',#798,#799,#800);
#795=PLANE('Hole Depth Surface',#801);
#796=TOLERANCED_LENGTH_MEASURE(8.0,$);
#797=THROUGH_BOTTOM_CONDITION();
#798=CARTESIAN_POINT('point for Placement for Hole',(87.5,80.0,0.0));
#799=DIRECTION('axis direction for Placement for Hole',(0.0,0.0,1.0));
#800=DIRECTION('reference direction for Placement for Hole',(1.0,0.0,0.0));
#801=AXIS2_PLACEMENT_3D('Axis for Placement for Hole Depth',#802,#803,#804);
#802=CARTESIAN_POINT('point for Placement for Hole Depth',(0.0,0.0,-40.0));
#803=DIRECTION('axis direction for Placement for Hole Depth',(0.0,0.0,1.0));
#804=DIRECTION('reference direction for Placement for Hole Depth',(1.0,0.0,0.0));
#805=PLANE('Drilling plane',#808);
#806=ROUND_HOLE('hole4',#3,(#807),#812,#813,#814,$,#815);
#807=DRILLING($,$,'Drilling',$,$,$,$,$,$,$,$,$,$,$);
#808=AXIS2_PLACEMENT_3D('Axis for Drilling plane placement',#809,#810,#811);
#809=CARTESIAN_POINT('point for Drilling plane placement',(0.0,0.0,10.0));
#810=DIRECTION('axis direction for Drilling plane placement',(0.0,0.0,1.0));
#811=DIRECTION('reference direction for Drilling plane placement',(1.0,0.0,0.0));
#812=AXIS2_PLACEMENT_3D('Axis for Placement for Hole',#816,#817,#818);
#813=PLANE('Hole Depth Surface',#819);
#814=TOLERANCED_LENGTH_MEASURE(8.0,$);
#815=THROUGH_BOTTOM_CONDITION();
#816=CARTESIAN_POINT('point for Placement for Hole',(50.0,86.0,0.0));
#817=DIRECTION('axis direction for Placement for Hole',(0.0,0.0,1.0));
#818=DIRECTION('reference direction for Placement for Hole',(1.0,0.0,0.0));
#819=AXIS2_PLACEMENT_3D('Axis for Placement for Hole Depth',#820,#821,#822);
#820=CARTESIAN_POINT('point for Placement for Hole Depth',(0.0,0.0,-40.0));
#821=DIRECTION('axis direction for Placement for Hole Depth',(0.0,0.0,1.0));
```

```
#822=DIRECTION('reference direction for Placement for Hole Depth',(1.0,0.0,0.0));
#823=PLANE('Drilling plane',#826);
#824=ROUND_HOLE('hole5',#3,(#825),#830,#831,#832,$,#833);
#825=DRILLING($,$,'Drilling',$,$,$,$,$,$,$,$,$,$,$);
#826=AXIS2_PLACEMENT_3D('Axis for Drilling plane placement',#827,#828,#829);
#827=CARTESIAN_POINT('point for Drilling plane placement',(0.0,0.0,10.0));
#828=DIRECTION('axis direction for Drilling plane placement',(0.0,0.0,1.0));
#829=DIRECTION('reference direction for Drilling plane placement',(1.0,0.0,0.0));
#830=AXIS2_PLACEMENT_3D('Axis for Placement for Hole',#834,#835,#836);
#831=PLANE('Hole Depth Surface',#837);
#832=TOLERANCED_LENGTH_MEASURE(8.0,$);
#833=THROUGH_BOTTOM_CONDITION();
#834=CARTESIAN_POINT('point for Placement for Hole',(11.0,51.0,0.0));
#835=DIRECTION('axis direction for Placement for Hole',(0.0,0.0,1.0));
#836=DIRECTION('reference direction for Placement for Hole',(1.0,0.0,0.0));
#837=AXIS2_PLACEMENT_3D('Axis for Placement for Hole Depth',#838,#839,#840);
#838=CARTESIAN_POINT('point for Placement for Hole Depth',(0.0,0.0,-40.0));
#839=DIRECTION('axis direction for Placement for Hole Depth',(0.0,0.0,1.0));
#840=DIRECTION('reference direction for Placement for Hole Depth',(1.0,0.0,0.0));
#841=PLANE('Drilling plane',#844);
#842=ROUND_HOLE('hole6',#3,(#843),#848,#849,#850,$,#851);
#843=DRILLING($,$,'Drilling',$,$,$,$,$,$,$,$,$,$,$);
#844=AXIS2_PLACEMENT_3D('Axis for Drilling plane placement',#845,#846,#847);
#845=CARTESIAN_POINT('point for Drilling plane placement',(0.0,0.0,10.0));
#846=DIRECTION('axis direction for Drilling plane placement',(0.0,0.0,1.0));
#847=DIRECTION('reference direction for Drilling plane placement',(1.0,0.0,0.0));
#848=AXIS2_PLACEMENT_3D('Axis for Placement for Hole',#852,#853,#854);
#849=PLANE('Hole Depth Surface',#855);
#850=TOLERANCED_LENGTH_MEASURE(8.0,$);
#851=THROUGH_BOTTOM_CONDITION();
#852=CARTESIAN_POINT('point for Placement for Hole',(106.0,21.0,0.0));
#853=DIRECTION('axis direction for Placement for Hole',(0.0,0.0,1.0));
#854=DIRECTION('reference direction for Placement for Hole',(1.0,0.0,0.0));
#855=AXIS2_PLACEMENT_3D('Axis for Placement for Hole Depth',#856,#857,#858);
#856=CARTESIAN_POINT('point for Placement for Hole Depth',(0.0,0.0,-40.0));
#857=DIRECTION('axis direction for Placement for Hole Depth',(0.0,0.0,1.0));
#858=DIRECTION('reference direction for Placement for Hole Depth',(1.0,0.0,0.0));
#859=PLANE('Security plane for pocket',#862);
#860=CLOSED_POCKET('pocket',#3,(#861,#866),#867,#868,(),$,#869,#870,#871,#872);
#861=BOTTOM_AND_SIDE_ROUGH_MILLING($,$,'Rough milling for pocket',$,$,$,$,$,$,$,$,$,$,$,$);
#862=AXIS2_PLACEMENT_3D('Axis for Security axis placement',#863,#864,#865);
#863=CARTESIAN_POINT('point for Security axis placement',(0.0,0.0,10.0));
#864=DIRECTION('axis direction for Security axis placement',(0.0,0.0,1.0));
#865=DIRECTION('reference direction for Security axis placement',(1.0,0.0,0.0));
#866=BOTTOM_AND_SIDE_FINISH_MILLING($,$,'Rough milling for pocket',$,$,$,$,$,$,$,$,$,$,$,$);
#867=AXIS2_PLACEMENT_3D('Axis for pocket placement',#873,#874,#875);
#868=PLANE('Pocket Depth plane',#876);
#869=PLANAR_POCKET_BOTTOM_CONDITION();
#870=TOLERANCED_LENGTH_MEASURE(8.0,$);
#871=TOLERANCED_LENGTH_MEASURE(0.5,$);
#872=GENERAL_CLOSED_PROFILE(#880,#881);
#873=CARTESIAN_POINT('point for pocket placement',(0.0,0.0,0.0));
#874=DIRECTION('axis direction for pocket placement',(0.0,0.0,1.0));
#875=DIRECTION('reference direction for pocket placement',(1.0,0.0,0.0));
#876=AXIS2_PLACEMENT_3D('Axis for Depth placement',#877,#878,#879);
#877=CARTESIAN_POINT('point for Depth placement',(0.0,0.0,-20.0));
#878=DIRECTION('axis direction for Depth placement',(0.0,0.0,1.0));
#879=DIRECTION('reference direction for Depth placement',(1.0,0.0,0.0));
#880=AXIS2_PLACEMENT_3D('Axis for profile placement',#882,#883,#884);
#881=POLYLINE('Pocket boundary',(#885,#886,#887,#888,#889));
#882=CARTESIAN_POINT('point for profile placement',(0.0,0.0,0.0));
#883=DIRECTION('axis direction for profile placement',(0.0,0.0,1.0));
#884=DIRECTION('reference direction for profile placement',(1.0,0.0,0.0));
#885=CARTESIAN_POINT('Point 1 for pocket polyline',(17.5,50.0,0.0));
#886=CARTESIAN_POINT('Point 2 for pocket polyline',(42.5,50.0,0.0));
#887=CARTESIAN_POINT('Point 3 for pocket polyline',(42.5,90.0,0.0));
#888=CARTESIAN_POINT('Point 4 for pocket polyline',(17.5,90.0,0.0));
#889=CARTESIAN_POINT('Point 5 for pocket polyline',(17.5,50.0,0.0));
#890=PLANE('Security plane for pocket',#893);
#891=CLOSED_POCKET('pocket',#3,(#892,#897),#898,#899,(),$,#900,$,$,#901);
```

```
#892=BOTTOM_AND_SIDE_ROUGH_MILLING($,$,'Rough milling for pocket',$,$,$,$,$,$,$,$,$,$,$,$);
#893=AXIS2_PLACEMENT_3D('Axis for Security axis placement',#894,#895,#896);
#894=CARTESIAN_POINT('point for Security axis placement',(0.0,0.0,10.0));
#895=DIRECTION('axis direction for Security axis placement',(0.0,0.0,1.0));
#896=DIRECTION('reference direction for Security axis placement',(1.0,0.0,0.0));
#897=BOTTOM_AND_SIDE_FINISH_MILLING($,$,'Rough milling for pocket',$,$,$,$,$,$,$,$,$,$,$,$);
#898=AXIS2_PLACEMENT_3D('Axis for pocket placement',#902,#903,#904);
#899=PLANE('Pocket Depth plane',#905);
#900=PLANAR_POCKET_BOTTOM_CONDITION();
#901=CIRCULAR_CLOSED_PROFILE(#909,#910);
#902=CARTESIAN_POINT('point for pocket placement',(0.0,0.0,0.0));
#903=DIRECTION('axis direction for pocket placement',(0.0,0.0,1.0));
#904=DIRECTION('reference direction for pocket placement',(1.0,0.0,0.0));
#905=AXIS2_PLACEMENT_3D('Axis for Depth placement',#906,#907,#908);
#906=CARTESIAN_POINT('point for Depth placement',(0.0,0.0,-20.0));
#907=DIRECTION('axis direction for Depth placement',(0.0,0.0,1.0));
#908=DIRECTION('reference direction for Depth placement',(1.0,0.0,0.0));
#909=AXIS2_PLACEMENT_3D('Axis for profile placement',#911,#912,#913);
#910=TOLERANCED_LENGTH_MEASURE(20.0,$);
#911=CARTESIAN_POINT('point for profile placement',(125.0,54.869,0.0));
#912=DIRECTION('axis direction for profile placement',(0.0,0.0,1.0));
#913=DIRECTION('reference direction for profile placement',(1.0,0.0,0.0));
#914=PLANE('Security plane for pocket',#917);
#915=CLOSED_POCKET('pocket',#3,(#916,#921),#922,#923,(),$,#924,$,$,#925);
#916=BOTTOM_AND_SIDE_ROUGH_MILLING($,$,'Rough milling for pocket',$,$,$,$,$,$,$,$,$,$,$,$);
#917=AXIS2_PLACEMENT_3D('Axis for Security axis placement',#918,#919,#920);
#918=CARTESIAN_POINT('point for Security axis placement',(0.0,0.0,10.0));
#919=DIRECTION('axis direction for Security axis placement',(0.0,0.0,1.0));
#920=DIRECTION('reference direction for Security axis placement',(1.0,0.0,0.0));
#921=BOTTOM_AND_SIDE_FINISH_MILLING($,$,'Rough milling for pocket',$,$,$,$,$,$,$,$,$,$,$,$);
#922=AXIS2_PLACEMENT_3D('Axis for pocket placement',#926,#927,#928);
#923=PLANE('Pocket Depth plane',#929);
#924=PLANAR_POCKET_BOTTOM_CONDITION();
#925=CIRCULAR_CLOSED_PROFILE(#933,#934);
#926=CARTESIAN_POINT('point for pocket placement',(0.0,0.0,0.0));
#927=DIRECTION('axis direction for pocket placement',(0.0,0.0,1.0));
#928=DIRECTION('reference direction for pocket placement',(1.0,0.0,0.0));
#929=AXIS2_PLACEMENT_3D('Axis for Depth placement',#930,#931,#932);
#930=CARTESIAN_POINT('point for Depth placement',(0.0,0.0,-20.0));
#931=DIRECTION('axis direction for Depth placement',(0.0,0.0,1.0));
#932=DIRECTION('reference direction for Depth placement',(1.0,0.0,0.0));
#933=AXIS2_PLACEMENT_3D('Axis for profile placement',#935,#936,#937);
#934=TOLERANCED_LENGTH_MEASURE(20.0,$);
#935=CARTESIAN_POINT('point for profile placement',(100.0,54.869,0.0));
#936=DIRECTION('axis direction for profile placement',(0.0,0.0,1.0));
#937=DIRECTION('reference direction for profile placement',(1.0,0.0,0.0));
#938=PLANE('Security plane for pocket',#941);
#939=CLOSED_POCKET('pocket',#3,(#940,#945),#946,#947,(),$,#948,$,$,#949);
#940=BOTTOM_AND_SIDE_ROUGH_MILLING($,$,'Rough milling for pocket',$,$,$,$,$,$,$,$,$,$,$,$);
#941=AXIS2_PLACEMENT_3D('Axis for Security axis placement',#942,#943,#944);
#942=CARTESIAN_POINT('point for Security axis placement',(0.0,0.0,10.0));
#943=DIRECTION('axis direction for Security axis placement',(0.0,0.0,1.0));
#944=DIRECTION('reference direction for Security axis placement',(1.0,0.0,0.0));
#945=BOTTOM_AND_SIDE_FINISH_MILLING($,$,'Rough milling for pocket',$,$,$,$,$,$,$,$,$,$,$,$);
#946=AXIS2_PLACEMENT_3D('Axis for pocket placement',#950,#951,#952);
#947=PLANE('Pocket Depth plane',#953);
#948=PLANAR_POCKET_BOTTOM_CONDITION();
#949=CIRCULAR_CLOSED_PROFILE(#957,#958);
#950=CARTESIAN_POINT('point for pocket placement',(0.0,0.0,0.0));
#951=DIRECTION('axis direction for pocket placement',(0.0,0.0,1.0));
#952=DIRECTION('reference direction for pocket placement',(1.0,0.0,0.0));
#953=AXIS2_PLACEMENT_3D('Axis for Depth placement',#954,#955,#956);
#954=CARTESIAN_POINT('point for Depth placement',(0.0,0.0,-20.0));
#955=DIRECTION('axis direction for Depth placement',(0.0,0.0,1.0));
#956=DIRECTION('reference direction for Depth placement',(1.0,0.0,0.0));
#957=AXIS2_PLACEMENT_3D('Axis for profile placement',#959,#960,#961);
#958=TOLERANCED_LENGTH_MEASURE(20.0,$);
#959=CARTESIAN_POINT('point for profile placement',(75.0,54.869,0.0));
#960=DIRECTION('axis direction for profile placement',(0.0,0.0,1.0));
#961=DIRECTION('reference direction for profile placement',(1.0,0.0,0.0));
```

```
#962=PLANE('Drilling plane',#965);
#963=ROUND_HOLE('New Hole',#3,(#964),#969,#970,#971,$,#972);
#964=DRILLING($,$,'Drilling',$,$,$,$,$,$,$,$,$,$,$);
#965=AXIS2_PLACEMENT_3D('Axis for Drilling plane placement',#966,#967,#968);
#966=CARTESIAN_POINT('point for Drilling plane placement',(0.0,0.0,10.0));
#967=DIRECTION('axis direction for Drilling plane placement',(0.0,0.0,1.0));
#968=DIRECTION('reference direction for Drilling plane placement',(1.0,0.0,0.0));
#969=AXIS2_PLACEMENT_3D('Axis for Placement for Hole',#973,#974,#975);
#970=PLANE('Hole Depth Surface',#976);
#971=TOLERANCED_LENGTH_MEASURE(0.0,$);
#972=THROUGH_BOTTOM_CONDITION();
#973=CARTESIAN_POINT('point for Placement for Hole',(0.0,0.0,0.0));
#974=DIRECTION('axis direction for Placement for Hole',(0.0,0.0,1.0));
#975=DIRECTION('reference direction for Placement for Hole',(1.0,0.0,0.0));
#976=AXIS2_PLACEMENT_3D('Axis for Placement for Hole Depth',#977,#978,#979);
#977=CARTESIAN_POINT('point for Placement for Hole Depth',(0.0,0.0,0.0));
#978=DIRECTION('axis direction for Placement for Hole Depth',(0.0,0.0,1.0));
#979=DIRECTION('reference direction for Placement for Hole Depth',(1.0,0.0,0.0));
#980=PLANE('Security plane for pocket',#983);
#981=CLOSED_POCKET('pocket',#3,(#982,#987),#988,#989,(),$,#990,$,$,#991);
#982=BOTTOM_AND_SIDE_ROUGH_MILLING($,$,'Rough milling for pocket',$,$,$,$,$,$,$,$,$,$,$,$);
#983=AXIS2_PLACEMENT_3D('Axis for Security axis placement',#984,#985,#986);
#984=CARTESIAN_POINT('point for Security axis placement',(0.0,0.0,10.0));
#985=DIRECTION('axis direction for Security axis placement',(0.0,0.0,1.0));
#986=DIRECTION('reference direction for Security axis placement',(1.0,0.0,0.0));
#987=BOTTOM_AND_SIDE_FINISH_MILLING($,$,'Rough milling for pocket',$,$,$,$,$,$,$,$,$,$,$,$);
#988=AXIS2_PLACEMENT_3D('Axis for pocket placement',#992,#993,#994);
#989=PLANE('Pocket Depth plane',#995);
#990=PLANAR_POCKET_BOTTOM_CONDITION();
#991=CIRCULAR_CLOSED_PROFILE(#999,#1000);
#992=CARTESIAN_POINT('point for pocket placement',(0.0,0.0,0.0));
#993=DIRECTION('axis direction for pocket placement',(0.0,0.0,1.0));
#994=DIRECTION('reference direction for pocket placement',(1.0,0.0,0.0));
#995=AXIS2_PLACEMENT_3D('Axis for Depth placement',#996,#997,#998);
#996=CARTESIAN_POINT('point for Depth placement',(0.0,0.0,-20.0));
#997=DIRECTION('axis direction for Depth placement',(0.0,0.0,1.0));
#998=DIRECTION('reference direction for Depth placement',(1.0,0.0,0.0));
#999=AXIS2_PLACEMENT_3D('Axis for profile placement',#1001,#1002,#1003);
#1000=TOLERANCED_LENGTH_MEASURE(20.0,$);
#1001=CARTESIAN_POINT('point for profile placement',(112.5,80.0,0.0));
#1002=DIRECTION('axis direction for profile placement',(0.0,0.0,1.0));
#1003=DIRECTION('reference direction for profile placement',(1.0,0.0,0.0));
#1004=PLANE('Security plane for pocket',#1007);
#1005=CLOSED_POCKET('pocket',#3,(#1006,#1011),#1012,#1013,(),$,#1014,$,$,#1015);
#1006=BOTTOM_AND_SIDE_ROUGH_MILLING($,$,'Rough milling for pocket',$,$,$,$,$,$,$,$,$,$,$,$);
#1007=AXIS2_PLACEMENT_3D('Axis for Security axis placement',#1008,#1009,#1010);
#1008=CARTESIAN_POINT('point for Security axis placement',(0.0,0.0,10.0));
#1009=DIRECTION('axis direction for Security axis placement',(0.0,0.0,1.0));
#1010=DIRECTION('reference direction for Security axis placement',(1.0,0.0,0.0));
#1011=BOTTOM_AND_SIDE_FINISH_MILLING($,$,'Rough milling for
pocket',$,$,$,$,$,$,$,$,$,$,$,$,$);
#1012=AXIS2_PLACEMENT_3D('Axis for pocket placement',#1016,#1017,#1018);
#1013=PLANE('Pocket Depth plane',#1019);
#1014=PLANAR_POCKET_BOTTOM_CONDITION();
#1015=CIRCULAR_CLOSED_PROFILE(#1023,#1024);
#1016=CARTESIAN_POINT('point for pocket placement',(0.0,0.0,0.0));
#1017=DIRECTION('axis direction for pocket placement',(0.0,0.0,1.0));
#1018=DIRECTION('reference direction for pocket placement',(1.0,0.0,0.0));
#1019=AXIS2_PLACEMENT_3D('Axis for Depth placement',#1020,#1021,#1022);
#1020=CARTESIAN_POINT('point for Depth placement',(0.0,0.0,-20.0));
#1021=DIRECTION('axis direction for Depth placement',(0.0,0.0,1.0));
#1022=DIRECTION('reference direction for Depth placement',(1.0,0.0,0.0));
#1023=AXIS2_PLACEMENT_3D('Axis for profile placement',#1025,#1026,#1027);
#1024=TOLERANCED_LENGTH_MEASURE(20.0,$);
#1025=CARTESIAN_POINT('point for profile placement',(87.5,80.0,0.0));
#1026=DIRECTION('axis direction for profile placement',(0.0,0.0,1.0));
#1027=DIRECTION('reference direction for profile placement',(1.0,0.0,0.0));
ENDSEC;
END-ISO-10303-21;
```

B.3. Heidenhain Listing

```
0  BEGIN PGM Aero MM
1  UNIT 700
2  BLK FORM 0.1 Z  X+0  Y+0  Z-50
3  BLK FORM 0.2  X+150  Y+100  Z+0
4  GLOBAL DEF100 COMMON ~
    Q200=+2     ;SET-UP CLEARANCE ~
    Q204=+50    ;2ND SET-UP CLEARANCE ~
    Q253=+750   ;F PRE-POSITIONING ~
    Q208=+99999 ;RETRACTION FEED RATE
5  GLOBAL DEF105 DRILLING ~
    Q256=+0.2   ;DIST FOR CHIP BRKNG ~
    Q210=+0     ;DWELL TIME AT TOP ~
    Q211=+0     ;DWELL TIME AT DEPTH
6  GLOBAL DEF110 POCKET MILLING ~
    Q370=+1     ;TOOL PATH OVERLAP ~
    Q351=+1     ;CLIMB OR UP-CUT ~
    Q366=+1     ;PLUNGE
7  GLOBAL DEF111 CONTOUR MILLING ~
    Q2=+1     ;TOOL PATH OVERLAP ~
    Q6=+2     ;SET-UP CLEARANCE ~
    Q7=+50    ;CLEARANCE HEIGHT ~
    Q9=+1     ;ROTATIONAL DIRECTION
8  GLOBAL DEF120 PROBING ~
    Q320=+0     ;SET-UP CLEARANCE ~
    Q260=+100   ;CLEARANCE HEIGHT ~
    Q301=+1     ;MOVE TO CLEARANCE
9  GLOBAL DEF125 POSITIONING ~
    Q345=+1     ;SELECT POS. HEIGHT
10 END OF UNIT 700
11 UNIT 232
12 TOOL CALL 1 Z S3000
13 CYCL DEF 232 FACE MILLING ~
    Q389=+2     ;STRATEGY ~
    Q225=-70    ;STARTNG PNT 1ST AXIS ~
    Q226=-20    ;STARTNG PNT 2ND AXIS ~
    Q227=+0     ;STARTNG PNT 3RD AXIS ~
    Q386=-5     ;END POINT 3RD AXIS ~
    Q218=+290   ;FIRST SIDE LENGTH ~
    Q219=+150   ;SECOND SIDE LENGTH ~
    Q202=+5     ;MAX. PLUNGING DEPTH ~
    Q369=+0     ;ALLOWANCE FOR FLOOR ~
    Q370= PREDEF ;MAX. OVERLAP ~
    Q207=+500   ;FEED RATE FOR MILLNG ~
    Q385=+500   ;FINISHING FEED RATE ~
    Q253= PREDEF ;F PRE-POSITIONING ~
    Q200= PREDEF ;SET-UP CLEARANCE ~
    Q357=+2     ;CLEARANCE TO SIDE ~
    Q204= PREDEF ;2ND SET-UP CLEARANCE
14 M3
15 CYCL CALL
16 END OF UNIT 232
17 UNIT 122
18 CONTOUR DEF ~
    P1 = "second.HC" P2 = "prime.HC"
19 TOOL CALL 0 Z S3000
20 CYCL DEF 20 CONTOUR DATA ~
    Q1=-15    ;MILLING DEPTH ~
    Q2= PREDEF ;TOOL PATH OVERLAP ~
    Q3=+0     ;ALLOWANCE FOR SIDE ~
    Q4=+0     ;ALLOWANCE FOR FLOOR ~
    Q5=+0     ;SURFACE COORDINATE ~
    Q6= PREDEF ;SET-UP CLEARANCE ~
    Q7= PREDEF ;CLEARANCE HEIGHT ~
    Q8=+0     ;ROUNDING RADIUS ~
    Q9= PREDEF ;ROTATIONAL DIRECTION
21 CYCL DEF 22 ROUGH-OUT ~
    Q10=-5     ;PLUNGING DEPTH ~
```

```
    Q11=+150  ;FEED RATE FOR PLNGNG ~
    Q12=+500  ;FEED RATE F. ROUGHNG ~
    Q18=+0    ;COARSE ROUGHING TOOL ~
    Q19=+0    ;FEED RATE FOR RECIP. ~
    Q208= PREDEF ;RETRACTION FEED RATE
22 M3
23 CYCL CALL
24 END OF UNIT 122
25 UNIT 125
26 SEL CONTOUR "inner1.HC"
27 TOOL CALL 0 Z S3000
28 CYCL DEF 270 CONTOUR TRAIN DATA ~
    Q390=+1    ;TYPE OF APPROACH ~
    Q391=+1    ;RADIUS COMPENSATION ~
    Q392=+5    ;RADIUS ~
    Q393=+90   ;CENTER ANGLE ~
    Q394=+0    ;DISTANCE
29 CYCL DEF 25 CONTOUR TRAIN ~
    Q1=-30   ;MILLING DEPTH ~
    Q3=+0    ;ALLOWANCE FOR SIDE ~
    Q5=+0    ;SURFACE COORDINATE ~
    Q7= PREDEF ;CLEARANCE HEIGHT ~
    Q10=-5    ;PLUNGING DEPTH ~
    Q11=+150  ;FEED RATE FOR PLNGNG ~
    Q12=+500  ;FEED RATE F. ROUGHNG ~
    Q15=+1    ;CLIMB OR UP-CUT
30 M3
31 CYCL CALL
32 END OF UNIT 125
33 UNIT 125
34 SEL CONTOUR "FistOut.HC"
35 TOOL CALL 0 Z S3000
36 CYCL DEF 270 CONTOUR TRAIN DATA ~
    Q390=+1    ;TYPE OF APPROACH ~
    Q391=+1    ;RADIUS COMPENSATION ~
    Q392=+5    ;RADIUS ~
    Q393=+90   ;CENTER ANGLE ~
    Q394=+0    ;DISTANCE
37 CYCL DEF 25 CONTOUR TRAIN ~
    Q1=-15   ;MILLING DEPTH ~
    Q3=+0    ;ALLOWANCE FOR SIDE ~
    Q5=+0    ;SURFACE COORDINATE ~
    Q7= PREDEF ;CLEARANCE HEIGHT ~
    Q10=-5    ;PLUNGING DEPTH ~
    Q11=+150  ;FEED RATE FOR PLNGNG ~
    Q12=+500  ;FEED RATE F. ROUGHNG ~
    Q15=+1    ;CLIMB OR UP-CUT
38 M3
39 CYCL CALL
40 END OF UNIT 125
41 UNIT 122
42 CONTOUR DEF ~
    P1 = "fist_in.HC"
43 TOOL CALL 0 Z S3000
44 CYCL DEF 20 CONTOUR DATA ~
    Q1=-10   ;MILLING DEPTH ~
    Q2= PREDEF ;TOOL PATH OVERLAP ~
    Q3=+0    ;ALLOWANCE FOR SIDE ~
    Q4=+0    ;ALLOWANCE FOR FLOOR ~
    Q5=+0    ;SURFACE COORDINATE ~
    Q6= PREDEF ;SET-UP CLEARANCE ~
    Q7= PREDEF ;CLEARANCE HEIGHT ~
    Q8=+0    ;ROUNDING RADIUS ~
    Q9= PREDEF ;ROTATIONAL DIRECTION
45 CYCL DEF 22 ROUGH-OUT ~
    Q10=-5    ;PLUNGING DEPTH ~
    Q11=+150  ;FEED RATE FOR PLNGNG ~
    Q12=+500  ;FEED RATE F. ROUGHNG ~
    Q18=+0    ;COARSE ROUGHING TOOL ~
    Q19=+0    ;FEED RATE FOR RECIP. ~
```

```
    Q208= PREDEF ;RETRACTION FEED RATE
46 M3
47 CYCL CALL
48 END OF UNIT 122
49 UNIT 252
50 PATTERN DEF ~
    POS1(  X+125  Y+54.869  Z+0 ) ~
    POS2(  X+100  Y+54.869  Z+0 ) ~
    POS3(  X+75  Y+54.869  Z+0 ) ~
    POS4(  X+112.5  Y+80  Z+0 ) ~
    POS5(  X+87.5  Y+80  Z+0 )
51 TOOL CALL 0 Z S3000
52 CYCL DEF 252 CIRCULAR POCKET ~
    Q215=+0    ;MACHINING OPERATION ~
    Q223=+20   ;CIRCLE DIAMETER ~
    Q368=+0    ;ALLOWANCE FOR SIDE ~
    Q207=+500  ;FEED RATE FOR MILLNG ~
    Q351= PREDEF ;CLIMB OR UP-CUT ~
    Q201=-20   ;DEPTH ~
    Q202=+5    ;PLUNGING DEPTH ~
    Q369=+0    ;ALLOWANCE FOR FLOOR ~
    Q206=+150  ;FEED RATE FOR PLNGNG ~
    Q338=+0    ;INFEED FOR FINISHING ~
    Q200= PREDEF ;SET-UP CLEARANCE ~
    Q203=+0    ;SURFACE COORDINATE ~
    Q204= PREDEF ;2ND SET-UP CLEARANCE ~
    Q370= PREDEF ;TOOL PATH OVERLAP ~
    Q366= PREDEF ;PLUNGE ~
    Q385=+500  ;FINISHING FEED RATE
53 M3
54 CYCL CALL PAT FMAX
55 END OF UNIT 252
56 UNIT 251
57 PATTERN DEF ~
    POS1(  X+30  Y+70  Z+0 )
58 TOOL CALL 0 Z S3000
59 CYCL DEF 251 RECTANGULAR POCKET ~
    Q215=+0    ;MACHINING OPERATION ~
    Q218=+25   ;FIRST SIDE LENGTH ~
    Q219=+40   ;SECOND SIDE LENGTH ~
    Q220=+0    ;CORNER RADIUS ~
    Q368=+0    ;ALLOWANCE FOR SIDE ~
    Q224=+0    ;ANGLE OF ROTATION ~
    Q367=+0    ;POCKET POSITION ~
    Q207=+500  ;FEED RATE FOR MILLNG ~
    Q351= PREDEF ;CLIMB OR UP-CUT ~
    Q201=-20   ;DEPTH ~
    Q202=+5    ;PLUNGING DEPTH ~
    Q369=+0    ;ALLOWANCE FOR FLOOR ~
    Q206=+150  ;FEED RATE FOR PLNGNG ~
    Q338=+0    ;INFEED FOR FINISHING ~
    Q200= PREDEF ;SET-UP CLEARANCE ~
    Q203=+0    ;SURFACE COORDINATE ~
    Q204= PREDEF ;2ND SET-UP CLEARANCE ~
    Q370= PREDEF ;TOOL PATH OVERLAP ~
    Q366= PREDEF ;PLUNGE ~
    Q385=+500  ;FINISHING FEED RATE
60 M3
61 CYCL CALL PAT FMAX
62 END OF UNIT 251
63 UNIT 251
64 PATTERN DEF ~
    POS1(  X+120  Y+90  Z+0 ) ~
    POS2(  X+70  Y+90  Z+0 ) ~
    POS3(  X+30  Y+90  Z+0 ) ~
    POS4(  X+30  Y+70  Z+0 ) ~
    POS5(  X+30  Y+50  Z+0 )
65 TOOL CALL 0 Z S3000
66 CYCL DEF 251 RECTANGULAR POCKET ~
    Q215=+0    ;MACHINING OPERATION ~
```

```
    Q218=+60   ;FIRST SIDE LENGTH ~
    Q219=+30   ;SECOND SIDE LENGTH ~
    Q220=+0    ;CORNER RADIUS ~
    Q368=+0    ;ALLOWANCE FOR SIDE ~
    Q224=+0    ;ANGLE OF ROTATION ~
    Q367=+0    ;POCKET POSITION ~
    Q207=+500  ;FEED RATE FOR MILLNG ~
    Q351= PREDEF ;CLIMB OR UP-CUT ~
    Q201=-15   ;DEPTH ~
    Q202=+5    ;PLUNGING DEPTH ~
    Q369=+0    ;ALLOWANCE FOR FLOOR ~
    Q206=+150  ;FEED RATE FOR PLNGNG ~
    Q338=+0    ;INFEED FOR FINISHING ~
    Q200= PREDEF ;SET-UP CLEARANCE ~
    Q203=+0    ;SURFACE COORDINATE ~
    Q204= PREDEF ;2ND SET-UP CLEARANCE ~
    Q370= PREDEF ;TOOL PATH OVERLAP ~
    Q366= PREDEF ;PLUNGE ~
    Q385=+500  ;FINISHING FEED RATE
67 M3
68 CYCL CALL PAT FMAX
69 END OF UNIT 251
70 UNIT 251
71 PATTERN DEF ~
    POS1(  X+20  Y+10  Z+0 )
72 TOOL CALL 0 Z S3000
73 CYCL DEF 251 RECTANGULAR POCKET ~
    Q215=+0    ;MACHINING OPERATION ~
    Q218=+60   ;FIRST SIDE LENGTH ~
    Q219=+40   ;SECOND SIDE LENGTH ~
    Q220=+0    ;CORNER RADIUS ~
    Q368=+0    ;ALLOWANCE FOR SIDE ~
    Q224=+0    ;ANGLE OF ROTATION ~
    Q367=+0    ;POCKET POSITION ~
    Q207=+500  ;FEED RATE FOR MILLNG ~
    Q351= PREDEF ;CLIMB OR UP-CUT ~
    Q201=-30   ;DEPTH ~
    Q202=+5    ;PLUNGING DEPTH ~
    Q369=+0    ;ALLOWANCE FOR FLOOR ~
    Q206=+150  ;FEED RATE FOR PLNGNG ~
    Q338=+0    ;INFEED FOR FINISHING ~
    Q200= PREDEF ;SET-UP CLEARANCE ~
    Q203=+0    ;SURFACE COORDINATE ~
    Q204= PREDEF ;2ND SET-UP CLEARANCE ~
    Q370= PREDEF ;TOOL PATH OVERLAP ~
    Q366= PREDEF ;PLUNGE ~
    Q385=+500  ;FINISHING FEED RATE
74 M3
75 CYCL CALL PAT FMAX
76 END OF UNIT 251
77 UNIT 205
78 PATTERN DEF ~
    POS1(  X+136  Y+86  Z+0 ) ~
    POS2(  X+112.5  Y+80  Z+0 ) ~
    POS3(  X+87.5  Y+80  Z+0 ) ~
    POS4(  X+50  Y+86  Z+0 ) ~
    POS5(  X+11  Y+51  Z+0 ) ~
    POS6(  X+106  Y+21  Z+0 )
79 TOOL CALL 3 Z S3000
80 CYCL DEF 205 UNIVERSAL PECKING ~
    Q200= PREDEF ;SET-UP CLEARANCE ~
    Q201=-40   ;DEPTH ~
    Q206=+150  ;FEED RATE FOR PLNGNG ~
    Q202=+5    ;PLUNGING DEPTH ~
    Q203=+0    ;SURFACE COORDINATE ~
    Q204= PREDEF ;2ND SET-UP CLEARANCE ~
    Q212=+0    ;DECREMENT ~
    Q205=+0    ;MIN. PLUNGING DEPTH ~
    Q258=+0.2  ;UPPER ADV STOP DIST ~
    Q259=+0.2  ;LOWER ADV STOP DIST ~
```

```
    Q257=+0    ;DEPTH FOR CHIP BRKNG ~
    Q256= PREDEF ;DIST FOR CHIP BRKNG ~
    Q211= PREDEF ;DWELL TIME AT DEPTH ~
    Q379=+0    ;STARTING POINT ~
    Q253= PREDEF ;F PRE-POSITIONING
81 M3
82 CYCL CALL PAT FMAX
83 END OF UNIT 205
84 UNIT 205
85 SEL PATTERN "drill.HP"
86 TOOL CALL 2 Z S3000
87 CYCL DEF 205 UNIVERSAL PECKING ~
    Q200= PREDEF ;SET-UP CLEARANCE ~
    Q201=-40   ;DEPTH ~
    Q206=+150  ;FEED RATE FOR PLNGNG ~
    Q202=+5    ;PLUNGING DEPTH ~
    Q203=+0    ;SURFACE COORDINATE ~
    Q204= PREDEF ;2ND SET-UP CLEARANCE ~
    Q212=+0    ;DECREMENT ~
    Q205=+0    ;MIN. PLUNGING DEPTH ~
    Q258=+0.2  ;UPPER ADV STOP DIST ~
    Q259=+0.2  ;LOWER ADV STOP DIST ~
    Q257=+0    ;DEPTH FOR CHIP BRKNG ~
    Q256= PREDEF ;DIST FOR CHIP BRKNG ~
    Q211= PREDEF ;DWELL TIME AT DEPTH ~
    Q379=+0    ;STARTING POINT ~
    Q253= PREDEF ;F PRE-POSITIONING
88 M3
89 CYCL CALL PAT FMAX
90 END OF UNIT 205
91 END PGM Aero MM
```